A PROPOSED REAL-TIME MONITOR AND DATA ANALYZER FOR AN AMERICAN COMMODITY EXCHANGE

by

Brian Smith Bentley

United States
Naval Postgraduate School

THESIS

A PROPOSED REAL-TIME MONITOR AND DATA ANALYZER FOR

AN

AMERICAN COMMODITY EXCHANGE

by

Brian Smith Bentley

June 1970

A Proposed Real-time Monitor and Data Analyzer

for an

American Commodity Exchange

by

Brian Smith Bentley
Lieutenant (junior grade), United States Naval Reserve
B.S., Tufts University, 1969

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SYSTEMS MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
June 1970

ABSTRACT

This Thesis considers the application of real-time digital information systems to the field of professional trading of commodities. Pertinent factors involved with the interfacing of a monitor system with a commodity exchange, and with professional trading interests, are considered. Skeletal functional outlines of factual and predictive displays are proposed from the monitoring of a general market environment to the monitoring of specific user positions in that environment. Elements of a monitoring system are implemented on an IBM 360/67 configured with an IBM 2250 Graphic Display Unit operating with OS/MVT. Due to the highly subjective nature of commodity decision-making, a scarcity of relevant published information in this area, and the author's lack of extensive trading experience, this thesis addresses itself only to the acquisition, summarization and presentation of objective information. Any presumption of data interpretation is left to the user.

TABLE OF CONTENTS

I. INTRODUCTION 5

II. SYSTEM OBJECTIVES 9

III. THE EXCHANGE INTERFACE 11

IV. THE USER INTERFACE 16

A. DISCUSSION OF "USEFUL INFORMATION" 16

B. DISCUSSION OF DISPLAY FORMAT 18

V. THE DATA MANAGEMENT SYSTEM 21

A. TRANSIENT CONSIDERATIONS 21

B. STEADY-STATE CONSIDERATIONS 23

C. ACTIVE MONITOR FUNCTIONS 29

D. NON-SESSION FUNCTIONS 30

VI. SUMMARY 31

VII. CONCLUSIONS 33

APPENDIX A: CHICAGO MERCANTILE EXCHANGE, STANDARD TRANSMISSION MNEMONICS 35

APPENDIX B: STATISTICAL FORECASTING 39

APPENDIX C: TEXT PRE-PROCESSOR 51

APPENDIX D: SAMPLE IMPLEMENTATION 64

BIBLIOGRAPHY 136

INITIAL DISTRIBUTION LIST 137

FORM DD 1473 139

Blank

I. INTRODUCTION

The commodity exchange approaches the economist's definition of a perfectly competitive market situation: prices are allowed to find an equilibrium level based solely upon what is believed to be the supply and demand. The exchange constitutes one of the most volatile marketplaces to be found anywhere in the world today. As a consequence, the opportunity to gain or lose huge amounts of capital exists daily. Such a situation attracts the speculator, and although the market does not specifically exist for such purposes, professional speculation serves a vital role in its function.

A commodity is generally taken to be a raw or semi-finished good or an argicultural or geological origin. Typical commodities include: grains-wheat, corn and oats, metals - copper, silver and platinum; meat - cattle, chicken and hogs; and diverse items such as orange juice, eggs and cocoa. Major industrial and agricultural entitles trade in these items, and many others, in more than a dozen special exchanges in the United States alone, trading on the order of hundred billion dollars per year. Price levels prevailing in these exchanges affect the American and even global economy at all levels from major corporations to the individual householder.

Among the giant entities for whose purposes the exchanges actually exist, there are to be found more than a million speculative individuals and groups. Through a process known as "hedging", these individuals accept the risks attendant to volatile price changes in return for the possibility of lucrative profit, and thereby act to insure the producer against unexpected future losses resulting fro adverse price movement.

The entity which must buy or sell physical materials through the exchange desires to guarantee the net price at which the ultimate transaction will occur, and is willing to forego profits in order to avoid losses.

The average speculator is willing to accept losses in order to accept the possibility of profit, and he is not as obviously protected as is the hedger. The speculator is lured by the theoretical possibility of returning hundreds of percent per day on his risked capital.

The basic vehicle of commodity speculation is the futures contract, which is, in essence, a binding promise to accept or deliver a warehouse receipt at some specified future date at a specified price, for a commodity which often does not yet physically exist. A futures contract is frequently measured in units of many tons of material, generally valued at the mean prevaling market price of several thousands dollars. For this reason, control of a futures contract is acquired at a fraction of its full value, usually on the order of ten percent, (several hundred to a few thousand dollars), depending upon the commodity and current market conditions.

The <u>average</u> price level of a commodity changes relatively slowly with time; however, the <u>instantaneous</u> price level is subject to minimal regulation, and is highly sensitive to factual and frequently fictional conditions of a highly diverse nature. As a consequence, the real price level can be subjected to extreme volatility during an active trading session. It is theoretically possible for the value of a contract to change by hundreds or even thousands of dollars in relatively few hours. Due to the high leverage afforded by exchange margin policies, the rate of return is considerable for the speculator who maintains a favorable position in the market. Conversely, the disfavored speculator sees his margin rapidly absorbed by an adverse price transient. At some predefined

point, the latter individual must commit additional capital to a transient of unknown amplitude, or choose to lose both control of his contract and all capital committed thusfar. In an especially volatile situation, this decision might present itself in a very short time after he has taken his initial position. If the speculator chooses to fight an adverse trend, he could commit a vast amount of capital to cover his position (e.g., a "short" position in a "bull" market). Should the futures contract eventually mature before the market conforms to his original expectations, his loss could prove considerable.

There exists an obvious motivation for the commodity speculator to stay abreast of market activities, virtually on a real-time basis. He must process considerable amounts of objective and subjective information of a highly complex nature, generated copiously by the one or more exchanges in which a sophisticated trader may be maintaining complex positions. While it is justifiably argued that successful speculating is largely intuitive, it is nevertheless true that much of the multiple data streams to which a speculator must expose himself could be preprocessed on a real-time basis, an obvious application for a digital computer.

The use of computers in commodity trading for speculative purposes is in its embryonic stages at best, and it is presently confined largely to attempts at long-term trending. Sources of machine compatible data are limited at present, both in terms of historical data and its depth and breadth.[1] The use of real-time systems is presently limited to two

[1]Standard and Poors of New York, long a supplier of such data to the securities industry, expects to release the first significant, reliable commodity data base in mid-1970.

major data networks,[2] whose systems basically perform the function the most recent state of a commodity or security for access via a basic display terminal.

The purpose of this thesis shall be the presentation of various factors affecting the implementation of a more generalized management information system, and the presentation of elements which would constitute such an implementation. Certain basic elements will be demonstrated in a functional system discussed in the appendices; this system has been implemented on an IBM 360/67 configuration with (2250) graphic display facility.

[2]Ultronics Systems Corporation and Bunker-Ramo Corporation, both based in New York, maintain such networks in the United States .

II. SYSTEM OBJECTIVES

The overall system has three functional objectives:

1. To capture coherent, accurate information online from a raw data stream, as it is officially transmitted from the commodity exchanges being monitored;

2. To store and display this information in an appropriate time frame of reference in a readily understood format, and,

3. To extend a forecast of expected market movement into the immediate future, presuming that short-term market movement is (to a degree) non-random; it may therefore be described in statistical, rather than economic, terms.

A system which will accomplish these objectives is proposed to consist of three elements:

1. an Exchange Interface, that boundary across which information is received from the monitored exchanges;

2. a User Interface, that across which information passes to and requests are received from the commodity trader; and,

3. a Data Management System, hereafter called "DMS", which acts to store and retrieve raw data, historical and predictive summaries and analyses, effect statistical forecasting, and process user requests for data display.

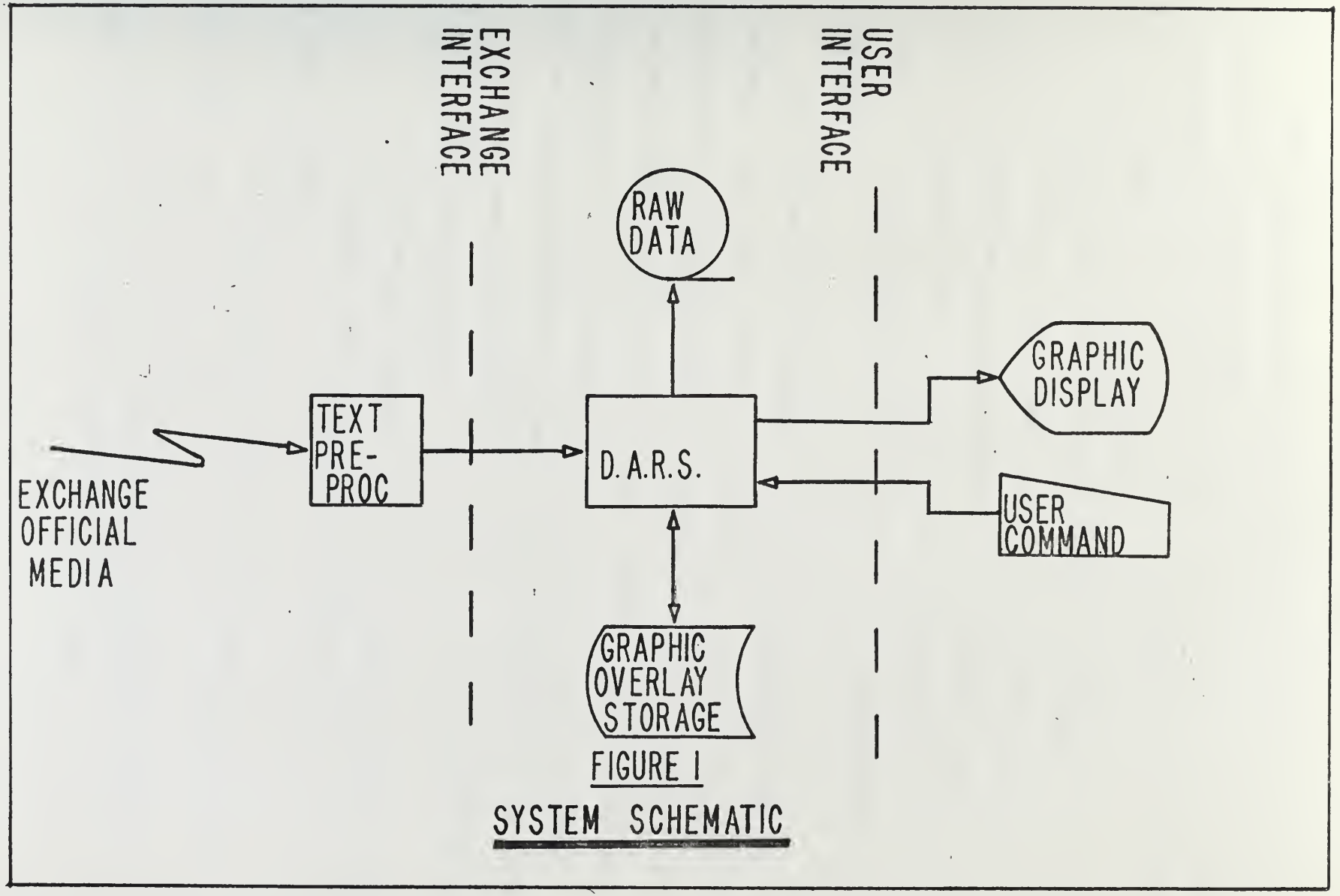

FIGURE 1

SYSTEM SCHEMATIC

III. THE EXCHANGE INTERFACE

The Exchange Interface is that boundary across which information passes to the DMS in a format suitable for the various analytic functions performed by the DMS. This data is received directly from the exchange being monitored, via a pre-processor.

Reporting practices of the various commodity exchanges are similar, but are not presented in a common format. For the sake of simplicity, only the Chicago Mercantile Exchange, one of the nation's largest commodity exchanges, will be discussed. Unless otherwise stated, all discussion of exchange practice refers to that of the Chicago Mercantile Exchange.

There are at least two methods of capturing information from the exchange. Both involve in-line pre-processors. The most direct method involves the use of a text-editor in addition to the other software of the DMS. Another method involves a remote pre-processing unit, perhaps serving many users, which actively transmits discrete units of information as they are received and extracted from the exchange reporting media.

A pre-processor should produce, as a discrete "exchange information unit", or "EIU", the following:

1. The commodity option in which the transaction has occurred;

2. The value of the transaction in standard units, specific to type of commodity;

3. Pertinent qualifiers, denoting the transaction as a bid, offer (ask), sale or volume report;

4. The exact time, designated to the appropriate fractional second, at which the transaction occurred at the exchange.

The exchange reports transactions as they occur, with a priority being given to the currently most-actively-traded options. This information is transmitted over 40-baud Western Union X1050 lines, using a special code. The transmission media has remained essentially the same for decades, and many receiving units, or "tickers" still actively in use were designed, if not built, in the 1930's.[3] Such receiving units present the data as it is being recorded at and transmitted from the exchange. Such presentation is in the form of an alphanumeric text stream, representing qualitative and quantitative exchange activity.

Human text editing is essentially too slow to interpret text for an automated system. It does have the advantage that it can smooth operator errors, more common in the earlier days of the transmission system. Although syntax errors are occasionally transmitted, the coding scheme appears highly standardized, and observed errors in sample text has been rare. Text-editing software should, therefore, be relatively straightforward.

Cursory software study indicates a necessary trade-off between the efficiency and the simplicity of the text-editor. Simple string pattern-matching is straightforward, but the necessity of repeated pattern-matching scans to recognize the many valid character pattern results in heavy overhead. A more efficient procedure involves a bottom-up parse of an assumed precedence grammer[4] in which the editor relates each character, as it is received, to those which it has already received.

[3]Western Electric Corporation, Description and Adjustments of the Typewheel Tape Printer, 1934.

[4]Feldman, J. and Gries, D., "Translator Writing Systems", CACM, Vol. 11, No. 2, p 77-113.

The system which performs the text-editing must also maintain a software-addressable clock for no other reason than to "time stamp" EIU's as they are extracted. In a small computer, real-time clocks are often available only at considerable additional expenses. Regardless of the type of the pre-processor, the EIU which it produces should contain at least the elements proposed above, in a standard format.

A separate pre-processing unit would appear to be more economical, shared among many users, than a configuration which must support both pre-processing and other DMS functions. The pre-processor must have the highest priority in any configuration which it shares with other functions. Since the pre-processors operation is essentially random, it must interrupt other problem processing as textual elements are received. If enough potential users do not exist to support a separate facility, or if the pre-processing function is sufficiently simple, then a user might find it optimal to maintain all functions, including the expense of a clock, in one configuration.

A possible source of pre-processed information might be those groups which presently operate real-time data banks for the securities industry. These networks gather and re-distribute data from many securities and commodity exchanges as it is produced. The data is stored on magnetic drums at regional data centers for access by Nixie-tube terminals and graphic display stations.[5] Little technical information was available to the author, but observation of their current mode of operation implies

[5]This was inferred by observation of an Ultronics satellite facility in San Francisco, California.

that there usefulness might be limited from the standpoint of a high-capacity analyzer. If the user must interrogate the regional data center for any large amount of data, especially on a continuous basis, then he might well find his system degraded through the request-queueing and message-handling procedure at the data center. Since, however, the regional data centers are being updated continuously by more central computer(s), the regional data center might maintain tables of users and information in which they are currently interested, and pass appropriate data directly to the user. For the purposes of this paper, however, three problems are as yet unresolved:

1. Data formats designed for special terminals may not be convenient as raw data for general processing. Special addressing data, graphic display orders and so forth may be only remotely related to the quantities which they represent.

2. Time information may be relatively-oriented ("time-since-last-transaction"), rather than absolutely-oriented ("time-of-last transaction"), which may prove inconvenient.

3. While securities exchanges maintain elaborate computer systems which may, in turn, feed the data network's distribution system, most commodity exchanges do not appear to do so. Since the network is then dependent upon the same ticker as is the user, one must examine the means (in terms of speed and accuracy) by which the network translates this text into distributable form.

Terminals are now being introduced which do perform certain active monitoring functions, one of which will monitor as many as eighteen separate issues and alert the user if user-set limits are exceeded.[6]

[6] This function is currently available on the Ultronics "Videomaster" graphic display terminal as an option.

This function is essentially instantaneous in time; the system does not maintain historical information. If tables of such limits are maintained at the regional center, as would seem proable, then it is not required that the network transmit actively to the terminal unless a breakout occurs, removing even that degree of activity.

IV. THE USER INTERFACE

The User Interface addresses two basic problems:

1. What constitutes useful information to the user; and,

2. What is the optimal display medium and format.

A. DISCUSSION OF "USEFUL INFORMATION".

The question of what constitutes useful information is highly subjective in any situation. A few years ago, Westinghouse Electric Corporation abandoned the idea of equipping all of its top-level executives with desk-side terminals presenting "real-time" information. The reason given was that "...few of the top managers could tell...what kind of information they wanted--or what they would do with it if they had it."[7] When a few veteran commodity traders were queried concerning the types of "real-time information" that they would like to see presented, the results were similar. It is difficult to draw even vague general conclusions about the industry, since, (as an authority on commodity trading notes:)

> "It is at least unfortunate that the commodity markets have had the skimpy attention given them by business writers to this point. Bookstore shelves groan under the weight of innumerable, well-executed volumes dealing with virtually every other facet of economics. But the seeker of useful information about commodity markets can carry the full selection available home under one arm. This, in spite of the fact that we are dealing with something in excess of a $100 billion annual trading volume."[8]

It becomes apparent that "useful commodity information" has yet to be defined objectively, especially information of a high interpretive

[7] Alexander, T., "Computers Can't Solve Everything", Fortune, V. 80, p. 128, October, 1969.

[8] Belveal, L., Charting Commodity Price Behavior, p. 3, Commodity Press, 1969.

nature. However, much remains to be done in the simple reporting and condensing of exchange data, and performing statistical extrapolations with such data with short enough delay time to be useful to an active trader. It is proposed that, in order of importance, an integrated system should present:

1. The immediate situation, i.e., that last transactions in relevent contracts as soon as possible after they occur. Many professional traders have direct telephone lines to the exchange floor to minimize this delay time.

2. The relevent historical situation, presented for comparison purposes in a readily understood format. This history should definitely include the current trading sessions transactions, and may include immediate access to the daily summary over the entire life of an option.

3. The expected situation for the remainder of the trading session, based primarily upon statistical considerations, presented in an easily understood format. Economic considerations are excluded, due to the relatively short time-span involved. It is expected that the experienced trader will be apprised of over-riding economic considerations in the market in which he is trading. It is also expected that he will make subjective allowance for these factors in interpreting all available data, including statistical expectations.

Simple systems which present a picture of the immediate situation, from the viewpoint of the general market, have been available for years. However, presentation of the market situation from the standpoint of a particular user is not so easy, especially from a human standpoint. In complicated market positions, such as "spreads" and "straddles", one is

frequently concerned by differences between two or more price trends. As the number of such complicated positions increases, one may find that he is soon limited by the number of positions which he can adequately watch. Presentation of such information, once his position(s) can be defined for a machine, is a relatively straightforward matter.

Presentation of the historical "market-oriented" situation is also fairly straightforward. One is simply displaying stored past data as received across the Exchange Interface. If the system has been consistently analyzing the market-oriented situation from the standpoint of the user's position(s), and storing this information, then a "user-oriented" situation may be readily presented.

Presentation of an expected situation can become quite complex in terms of the possible statistics involved. The availability of real-time analysis suggests certain validity to the assumption that statistical extrapolation alone is sufficient for a first-approximation, short-range forecast. If one imparts a certain degree of "inertia" to a market as complex as a commodity exchange, and if the lead time of the forecast is sufficiently short, then a statistical approximation may prove at least useful to the user.[9] (Human psychology and economics are, to be sure, critical market factors. However, modeling their effects objectively would prove extremely difficult, and would result in a heuristic evaluation process, which for the sake of brevity is not explored here.)

B. DISCUSSION OF DISPLAY FORMAT.

In this business, one appears not as interested in permanence of information as he is in its rapid and reliable availability. Since decisions appear to be required in "real time" there is nothing more dead than the detailed transactions on yesterday's market, except to the degree

[9] See Appendix B.

that they indicate those of the rest of today. A paper tape ticker frequently falls directly into a wastebasket. The ticker is reliable and fast, and presents a crude information retrieval system, which is why it has been in use for so long amid increasingly sophisticated information media. One might conclude that hardcopy is a convenient by-product, not essential in itself.

An ideal information display device is a cathode ray tube, or graphic display terminal. The most common terminals available today are limited to textual display. Such units limit one to alphanumeric displays, and therefore to a digital rather than analog display. If one must display information symbolically, then he must confine himself to the symbol set which is readily understood by the user. The symbolism required to describe historical and expected trends, especially if complex statistical method are involved, may not be familiar to the traditional user. If the statistical significance of such reductions is not readily and accurately understood, then the system may well be ignored in the pressure of daily trading.

The old adage, "A picture is worth a thousand words", is especially true in this general situation, and therefore dictates some form of analog, or graphic representation. Graphic display units with a vector-generating capability have generally been far more expensive than those which merely generate characters, selling from $8,000 to over $100,000 per unit.[10]

[10]The implementation on which this system was demonstrated consisted of an IBM 360/67 configured with a 2250-1 graphic display unit. The latter element reatils for over $100,000 and the overall installation above $4 million. However, Textronix, Inc., of Beaverton, Oregon, to cite one example, has introduced an essentially similar display unit for $8,000. Viatron Computer Systems Corp., of Bedford, Massachusetts, again to cite but one example, retails sufficiently sophisticated computers in the $4,000 to $10,000 range to perform all functions described in this thesis. Such prices are typical of trends in the industry.

Prices for these units are rapidly decreasing, and under lease system, implementation costs could prove quite reasonable in the near future.

The concept of an optimal display format is largely subjective, one which has been implemented for the purposes of this thesis consists of a resident image, which permanently displays vector elements common to all displays, with two types of graphic overlays:

1. An overlay containing all historical graphic and alphanumeric information. Historical graphs depict all the current session's transaction and volume reports. Alphanumeric information specifies the last transaction received, its type, numerical value and time of arrival.

2. An overlay containing all predictive graphic and alphanumeric information. Predictive graphs depict an extrapolated expected market trend based upon all transactions received to the present, projected forward from the time of the last received transaction. Alphanumeric information depicts significant parameters in statistical equations, which may be modified by the user, if necessary, to change the response rate and stability of the model.

The above representation permits only one option to be displayed at one time. Existing text-display terminals display a number of options of interest to the user simultaneously. However, the amount of information displayed is limited. Such a display is useful, and its inclusion is relatively straightforward.

V. THE DATA MANAGEMENT SYSTEM

The Data Management System functions in basically two states:

1. A Transient State, during periods of initiallization and shutdown; and,

2. A Steady State, which is considered to be the normal mode of operation.

A. TRANSIENT CONSIDERATIONS

The Transient State at initiallization can be classes as:

1. A "coldstart" situation, in which the system is being initiallized before the trading session opens; and,

2. A "warmstart" situation, in which the system must be restarted during a trading session.

The "coldstart" situation is relatively straightforward for two reasons:

First, both the Exchange and User Interfaces are essentially inactive, placing no time or priority processing constraints upon the system. Initiallization of system tables and parameters may occur at the leisure of both the system and the user.

Second, there is no information in the system which might be immediately required, or which requires immediate or completed processing.

The "warmstart" situation is more critical. Since it is defined to occur during a trading session, it is likely that both the Exchange and User Interfaces will be highly active, regardless of whether the system is functional. The pre-processor is transmitting to the DMS interface buffer and if some standby arrangement does not provide for storage of

this information (required to continue trend analysis), it will be lost. If the pre-processor is remote to the system, then the facility may exist to request lost information. If the pre-processor is an integral part of the DMS, then the "raw" text stream will be lost, and at this time, no facility exists to request "backup transmissions" over the ticker line. In practice, the activity on the Exchange Interface is not under control of the DMS or the user. Activity on the User Interface is obviously under control of the user; however, this is small consolation when the system is not providing vital information. In a "warmstart" situation, there must exist both a means of rapidly bootstrapping the DMS, and sufficient backup storage to re-initiallize resident core-storage tables and resident graphic buffers. Since graphic overlay files are resident on direct-access units, a large fraction of the system's information will not be damaged.

The Transient State at shutdown can be classes as:

1. A normal system termination, which occurs after the trading session is over; and,

2. An abnormal system termination, which occurs during the trading session through hardware or software failure.

The normal termination is straightforward, as it permits the system sufficient time to save pertinent information at its leisure. Since it will presumably be followed by a "coldstart", there is no requirement that information be saved in such a manner as to permit rapid re-starting of the system.

An abnormal termination is a more critical situation. Since a "warm-start" should follow this event, all considerations discussed above apply. An overriding factor is the length of time the system will take to "catch up" with the market once it is restarted, and this governs the amount of

information which must be stored as backup. Most of the system's efforts up until abnormal termination have involved statistical analysis and updating of graphic overlay files. Since these functions occur with the reception of each "information unit" from the pre-processor, and since such information resides on direct-access devices, it will be current for all monitored options, and will not be lost upon termination. The DMS should be designed such that core-resident tables are essentially static, serving to address direct-access files upon which all dynamic option-specific information is stored. Such tables may be restored from backup copies on direct-access files. Resident graphic displays should contain minimal dynamic information that cannot be immediately updated by incoming EIU's, or by access to resident files. Such information includes the last transaction for an option, and the current time.

All routines used in the transient state may be overlaid in core as called by the DMS.

B. STEADY-STATE CONSIDERATIONS

In Steady-State operation, the system functions at one of eight Priority Levels, each encompassing one task. At each Priority Level, the DMS maintains a Task Status Block, which contains pertinent information about the task. This block permits a task to be interrupted and restarted by the DMS Resident Monitor, should transfer of control to other Priority Levels become necessary. The DMS Resident Monitor, should transfer of control to other Priority Levels become necessary. The DMS Resident Monitor processor interrupts from the Exchange and User Interfaces, and passes control to appropriate interrupt levels. It provides any complex scheduling other than sequential, that a more advanced DMS might require.

The Priority Levels are summarized as follows; in order of urgency:

1. Processing interrupts from the Exchange Interface;
2. Processing interrupts from the User Interface;
3. Updating historical graphic overlay files;
4. Analysis of user's position relative to current market;
5. Updating predictive graphic overlay files;
6. Analysis of user's position relative to "expected" market;
7. Recommended "optimal" market positions (optional); and,
8. Idle state.

1. Priority Level One: Exchange Interface

Priority Level one is assigned to the Exchange Interface. This priority level pre-empts all other DMS functions because the EIU from the pre-processor must be transferred from the DMS interface buffer before the arrival of the next EIU. This Level may be entered only upon receipt of a hardware interrupt from the Exchange Interface of the DMS.

The task will perform the following functions:

a. Transfer EIU from interface buffer to sequential backup storage;

b. Extract "time-stamp" from EIU. If the DMS requires the current time for display or other reasons, this should be within a few seconds of the correct time.

c. If the DMS is not monitoring this option, return control to Resident Monitor. This might be quickly determined by "hash-addressing" the commodity option mnemonic, and checking a suitable table-entry for its presence. Such a table might also contain direct access file information for other tasks.

d. If the option is being monitored, then its internal identification is its hash-code.[11] Convert any qualifiers and numerical representation in the EIU to internal representation, and return control to the Resident Monitor.

Priority Level one may not be interrupted during execution, except by system failure.

2. Priority Level Two: The User Interface.

Priority Level two is assigned to the User Interface. It is given control when the user requests one of the following three actions:

a. A change in the active graphic display;

b. A change in those system parameters which may be dynamically modified; or,

c. A change in the user's specific position.

Level two may be entered only by an interrupt from the User Interface, and may be pre-empted while in excecution only by Level one. Upon user command, this task will:

a. Retrieve appropriate graphic overlays from direct-access storage and display them from the graphic display unit buffer, whenever a display change is required.

[11] A "hash-address" or "hash code" involves the use of a data element's structure to calculate a unique table or file address for an elements of that exact structure. In this demonstrated implementation, a hash code is generated by adding together the integer representations of the EBCDIC characters in the option mnemonic for a commodity, and performing a modulo-19 operation, generating an address between 0 and 19 inclusive. This operation is much faster than scanning a table of nineteen entries for a matching character pattern. The savings becomes far more significant if the system must rapidly compare a mnemonic against dozens or even hundreds of valid patterns. This technique for generating hash addresses is one of many. See Morris, R., "Scatter Storage Techniques", CACM, Vol. 11, No.11, p. 38-44.

b. Update appropriate table and graphic files, re-initiallizing statistical data as necessary whenever a system parameter is modified.

c. Service requests for a change in the user's specific position by allocating or releasing core and direct-access storage devoted to the affected positions, and initiallizing analysis of new positions with respect to the market.

This task is given second level priority for two reasons:

a. In most cases, only a change in display will be requested. If complete graphic overlays can be paged as one physical record, then execution of these requests will require relatively few machine cycles. It is presumed that once the graphic overlay is in the display unit buffer, the DMS is finished with the request.

b. The primary purpose of the system is to display and maintain information of immediate use to the user.

3. Priority Level Three: Historical Graphic Update General

Priority Level Three is assigned the task of updating relevent historical graphic overlay files. The Resident Monitor of the DMS transfers control to this level following execution of a level-one task if the level-two task is not pending completion. This task performs the following functions:

a. Find, load into core and update the relevent historical overlay.

b. If this overlay is being actively displayed, transfer a copy of the overlay to the display unit buffer.

c. Transfer the overlay back to direct-access storage, and return control to Resident Monitor.

This task should execute relatively fast if:

a. The overlay is stored upon direct-access device as one physical record, and may be transferred as such; and,

b. Graphic orders are generated by core-resident routines.

4. Priority Level Four: Historical Graphic Update, Specific

Priority Level Four essentially analyzes all user positions involving that option reported upon by the last EIU and updates graphic files describing each position accordingly. This task executes in a manner similar to that at Level Three.

Constraints affecting the efficiency of this task are similar to those of Level Three, with an added constraint relating to the number of positions which must be analyzed. Since information is merely added to appropriate graphic order files, as in Level Three, the process should be relatively efficient.

5. Priority Level Five: Predictive Graphic Update, General

Priority Level Five is assigned the task of updating relevant predictive graphic overlay files. It performs the following functions:

a. Find, load into core and extract statistical data stored with file.

b. Using statistical data, recursively determine new statistics and replace in record image. Utilize new statistics to determine co-efficients for forecast model equation.

c. Regenerate entire graphic overlay file from "current" point on overlay's time axis, using forecast equation.

d. If overlay is being actively displayed, transfer a copy of overlay to display device buffer.

e. Transfer record image back to direct-access storage, and return control to Resident Monitor.

The efficiency of this operation is subject to the same constraints as Level Three, with the added constraint that the type of forecasting and the amount of calculation necessary to generate forecast points is important.

This task is placed at level five because it is felt that statistical prediction of market trends is subordinate to factual current market data, especially to one who is maintaining an active position.

6. Priority Level Six: Predictive Graphic Update, Specific.

Priority Level Six analyzes each relevant market position, as in Level Five, for its expected net value in light of the new information. Each overlay file must maintain updated co-efficients for the forecast equations of each commodity option involved in the position; these co-efficients must be updated for the specific option involved. The net expected value of the position as a function of time is in turn a function of these equations, plus other pertinent data such a commissions, carrying charges and so forth. The expected position must be regenerated from the "current" point on the time axis of the display, using this equation, in a manner similar to Level Five. If the system is actively monitoring price limits, then it would determine the time at which the limits will be reached, if it appears that they will be reached during the current session.

7. Priority Level Seven: Strategy Recommendation.

Level Seven examines the expected movement of all monitored commodity options to suggest optimal market positions, especially

in more complex areas such as spreads or straddles. The discussion of this option is limited to that of a "blue-sky" suggestion since its implementation would be complex. However, the information does exist, following completion of the task at Level Five, to determine optimal strategies based upon expected movement.

8. Priority Level Eight is essentially an idle state, at which the Resident Monitor awaits higher level interrupts.

Scheduling among priority levels is essentially sequential, in which control is passed to the next lower priority level as higher level functions are completed. An obvious danger with this type of scheduling results if Exchange and the inter-arrival time of or User interrupts is shorter than the time required for the system to naturally progress to lower levels. Priority Levels below which the system, on the average, has time to reach, will never be completed. Therefore, the Resident Monitor may have to determine, on the basis of mean inter-arrival rate, and mean task completion time, exactly how low in its priority-level hierarchy it will schedule tasks. The scheduling can then become dynamic and quite complex.

C. ACTIVE MONITOR FUNCTIONS

As discussed above, the DMS performs a basically passive function, compiling and presenting graphic data on request. The DMS could play an active role by accepting limits of interest to the user, and notifying the user when:

1. It is expected that the limits will be exceeded in the current session, and when those limits will be exceeded; and

2. Limits are actually exceeded in trading.

This function could be performed for both the general market, and for specific user positions. Such active functions are implemented on commercial system now, although only in the sense of (2) above. They relieve the user of much of the effort of constantly watching tapes and displays and trying to guess the significance of trends.

D. NON-SESSION FUNCTIONS

A commodity exchange is active a relatively small percentage of the time. When the exchange is active, the full resources of the system may be devoted to some or all of the functions in (B) above. However, there are at least twenty hours a day when the exchange is not active, and during which the system may be devoted to other functions.

An obvious function involves optimization of statistical methods or parameters used during active session, based upon "raw" EIU's stored at Priority Level One during previous sessions.

Another function involves statistical processing of longer-range data, searching for correlations between such diverse fundamental factors as weather, crop reports, unused stocks and so forth, and such technical factors as volume and open interest trends. Such information might be summarized as graphic displays which may be paged from direct-access file for reference to the longer-term fundamental or technical situation. Forecasts for several days might be available to the user in interpreting the significance of emerging trends during a trading session.

VI. SUMMARY

In summation, it has been proposed that a functional digital information system may be composed of

1. An interface with the official media of specific exchanges, through a pre-processing unit which converts various exchange data formats to a standardized format, which includes time of origin.

2. An interface with the user through an interactive vector-generating graphic display terminal, by which he may request suitably summarized information concerning the market environment and his specific position in that environment.

3. An operating system which maintains graphic order files on direct-access storage of an historical (e.g., factual) and predictive nature, as a series of user-callable graphic overlays, performing suitable statistical and graphic updates as new information is received.

The author does not possess the extensive trading or economic knowledge which would be required to suggest specific online interpretive functions of a heuristic rather than statistical nature. They would be the subject of much more sophisticated work in the embryonic application of information technology. It might be noted that any heuristic system which attempted to determine market patterns by observation of large masses of data would find such data to be largely non-existent, except in a daily summary of minimal detail. Previously demonstrated calculations imply that offline storage requirements for the information transmitted by the official exchange media are not excessive. It would appear that profitable work remains to be done in the fields of simply getting useful, machine compatible information to potential users. A

secondary, but ultimate purpose involves the presentation of better information to the professional trading interests in sufficient time to be of use in market decisions.

VII. CONCLUSIONS

A number of final observations have resulted from work thusfar concerning a Monitor System.

Since volume reports arrive once every half hour, it would appear that volume predictions would be of limited accuracy or usefulness. It would further appear that the volume of data is sufficiently low that a mere factual presentation is sufficient, and that the user should draw appropriate conclusions from this data alone. Considerable work remains to be done designing a text editor which is capable of processing a general text stream for a particular exchange.

The problems involved in statistical forecasting (aside from any economic or psychological considerations) have received but cursory attention in this thesis. "Exponential smoothing" is not the only feasible approach to statistical forecasting; some might maintain that it is far from the best. Even this form of smoothing has not been exhaustively treated in this work. Study remains in the areas of minimizing both variance and confidence intervals for projections, which this implementation does not do, and conveying this information to the user, perhaps as a graphically-displayed "confidence envelope" about a projected forecast.

Proposed in a general manner has been the subject of analysis of specific user positions in a market. There remains the detailed design of software to permit the entry, deletion, modification and storage of user positions, optimized from the standpoint of time and storage. Attendant to this is the summarization of all factors affecting the net value of various types of positions, including carrying charges, brokerage

and exchange fees, and so forth. A useful interactive facility, if the system's excess capacity permitted, would permit the user to study the net present and expected value of shifts in positions and combinations thereof, consistent with charges and fees such as those mentioned above.

APPENDIX A

CHICAGO MERCANTILE EXCHANGE
STANDARD TRANSMISSION MNEMONICS

Approximately five major types of information are transmitted over the official Chicago Mercantile Exchange line, as standardized mnemonic codes. These are as follows:

A. CONTRACT OPTION DESIGNATION.

A futures contract option (maturity date of a futures contract) is designated through the concatenation of a one-or two-character mnemonic indicating the specific option, and a one-character mnemonic indicating the month of maturity of the contract. Important standard mnemonics are:

COMMODITY		MATURITY	
Butter	B	January	F
Eggs	E	February	G
Live Cattle	LC	March	H
Live Hogs	LH	April	J
Frozen Pork Bellies	PB	May	K
Idaho Potatoes	P	June	M
		July	N
		August	Q
		September	U
		October	V
		November	X
		December	Z

For example, February Egg futures would be designated as "EG", and March Live Cattle futures would be designated as "LCH".

In cases in which the year of maturity is not immediately obvious, the year will be concatenated onto the aggregate mnemonic defined above.

B. ACTION SPECIFICATION

An action is specified by a numerical value, preceeded by a qualifier,

if appropriate. It refers to the last futures contract option designated in the data stream.

1. Numerical Value.

The numerical value generally consists of an integer portion concatenated with a two-place decimal fraction. Its unit of measurement is in hundredths of a dollar (i.e., in cents), and is a price attached to one standard specific unit (e.g., pounds, troy ounces, bushels, etc.), of the designated futures option. (It is not the price of one full contract, the minimum trading unit, which may be defined in terms of thousands of such specific standard units). The numerical value specifies this price to the nearest one-hundredth of a cent (or "point"). In certain instances, the fractional portion may be transmitted as one of seven rational fraction, expressed in increments of one-eighth of a cent. This has not been observed in normal transmission of trading actions, however, but rather to report summary data.

For example, a reference to an option of pork bellies (raw bacon futures) at thirty-five cents per pound would be transmitted as "35.00" in the data stream.

2. Qualifiers

A qualifier may consist of one of four types, and indicates the nature of the numerical value which immediately follows it.

a. A completed sale is indicated by the absence of any qualifiers, rather, the numberical value of the price at which the sale occurred is transmitted.

b. A bid (an offer to buy) ("bid") tendered at a price given by the numerical value is indicated by the character "B" immediately preceeding that numerical value.

c. An offer to sell ("ask") at a price indicated by the numerical value is transmitted with the character "A" immediately preceeding that numerical value.

Numerical values, qualifiers and futures option designations are usually separated by two or more dots in cases where confusion might otherwise result. Moreover, characters are printed in one of two parallel text lines on the tape, depending upon the nature of the character, for ease in reading the tape. The transmitted data for a hypothetical text stream might be:

"....PBH...43.27...43.30...A..44.00...B.43.20...43.25...".

This data stream would indicate that all further data would refer to the March option of Frozen Pork Bellies (PBH), until another valid "contract option designation" occurred. A completed sale is reported at 43.27 cents per pound (for an unspecified number of contracts), and another sale is reported at 43.30 cents per pound. A prospective seller then offers to buy at 43.20 cents per pound. Another sale is reported at 43.25 cents per pound.

There is no pre-defined order in which market actions are reported, nor may all market actions be reported. A transmission priority is given to the currently most active options.

C. VOLUME SPECIFICATIONS

Every half hour, between 10:00 and 1:00 p.m. inclusive, the Chicago Mercantile Exchange reports cumulative volume figures broken down by specific options for the current trading session. This information is preceeded by the mnemonic ".SALES.TO" in the data stream followed by the

time in one to four digits, as appropriate. The succeeding data stream then consists of contract option designations and integral cumulative sales figures (for that session). This series is broekn by two or more dots, which might be taken by an algorithmic text editor to terminate the volume report.

D. TIME DISIGNATIONS.

Every five minutes, the Chicago Mercantile Exchange transmits the current Chicago time. The time designation consists of one to four digits, as appropriate, concatenated to the mnemonic "AM." or "PM.", again as appropriate.

E. CORRECTION DESIGNATIONS

Two types of corrections have been observed in sample data streams:

1. Previous Entry Deletion

The following mnemonic pattern is transmitted to delete previous entries in the data stream:

"CANCEL.LAST" + xxx + yyyyyy,

in which, xxx represents the contract option designation; and, yyyyyy represents the numeric value and qualifier (if any) of the option to be deleted.

2. Previous Entry Substitution

The following mnemonic pattern is transmitted to correct previous erroneous entries in the data stream:

"MAKE.LAST" + xxx + yyyyyy + "READ" + zzzzzz,

in which, xxx and yyyyyy are defined as above; and, zzzzzz represents the corrected value to be substituted.

APPENDIX B

STATISTICAL FORECASTING

A. DISCUSSION OF THEORY

Statistical forecasting is based, to an extent, upon the assumption that any dynamic physical process may be described in its relevant dimensions by purely mathematical models. Given a suitable model, all of its parameters may be determined by observation of the original process. Once those parameters have been determined, then the future behavior of the process is presumed to be described by the model. For the purposes of this thesis, the problem is three-fold:

1. The determination of dynamic state equations of the process as a function of all relevant dimensions; since this model is primarily time-dependent, it will be assumed that the model is described as a function of time only.

2. The employment of suitable statistical techniques to determine co-efficients of the model. The technique should presume that the co-efficients themselves may vary slowly with time as the process continues, or at least approach a theoretical limit asymptotically, and utilize new data as it arrives to maintain convergence.

3. In a computer environment, both time and fast-access storage are limited, the former by the interarrival time of data, and the latter by the less pressing, but real, cost of core storage. There is, therefore, an incentive to limit the amount of historical data which must be stored and continuously reprocessed, while still maintaining a practical level of accuracy in generated forecasts. This consideration limits the application of exotic curve-fitting techniques, which can require considerable

amounts of processing time, and most of the relevant historical data. Since data tends to accumulate with time, storage limitations can become significant. One might presume that the relative significance of data decreases with its increasing age, and the availability of new data. It would therefore seem preferable to accumulate the statistical significance of past data in a relatively small amount of storage, and discount all past data as an aggragate as new data arrives. The statistical process then becomes essentially recursive, in which new aggregate statistics become a function of new data and previous aggregate statistics.

A standard technique, known as "exponential smoothing", employs these principals. Rather than present a derivation of this technique, a summary of relevant models is presented:

1. Single-Exponential Smoothing, a constant Model.

The constant model presumes that the monitored process is essentially constant in time, and is described by the equation:

$$E(t) = a_0 .$$

The smoothing statistic is single-exponential:

$$S_t^{(1)}(x) = \alpha[x(t)] + \beta[S_{t-1}^{(1)}(x)]$$

where,

$$\beta = 1 - \alpha$$

and,

$$S_0^{(1)}(x) = a_0(0) \qquad \text{(essentially an estimate),}$$

and x(t) is an observation taken at time, t. The "smoothing constant, α,

ranges from 0 to 1 and governs the decay rate of past observations. Observations are presumed to be equally spaced in time. A forecast of time periods from time, T, projected at time, T, is given by,

$$\hat{X}_\tau(T) = S_T(X)$$

2. Double-Exponential Smoothing, a linear model

The linear model presumes that the monitored process is subjected to constant trends in time, and is described by the equation:

$$E(t) = a_0 + a_1 t$$

The smoothing statistics are double-exponential:

$$S_t^{(1)}(X) = \alpha[X(t)] + \beta[S_{t-1}^{(1)}(X)]$$
$$S_t^{(2)}(X) = \alpha[S_t^{(1)}(X)] + \beta[S_{t-1}^{(2)}(X)]$$

Initial conditions for the statistics are:

$$S_0^{(1)}(X) = a_0(0) - \frac{\beta}{\alpha} a_1(0)$$
$$S_0^{(2)}(X) = a_0(0) - \frac{2\beta}{\alpha} a_1(0)$$

a forecast made at time, T, for periods ahead in time is given by:

$$\hat{X}_\tau(T) = \left(2 + \frac{\alpha\tau}{\beta}\right)[S_T^{(1)}(X)] - \left(1 + \frac{\alpha\tau}{\beta}\right)[S_T^{(2)}(X)]$$

3. Triple-Exponential Smoothing, a quadratic model

The quadratic model presumes that the monitored process is subjected to a constantly-changing trend in time (which may, in fact, be zero, the trivial reduction to the linear case). The model is described by the equation:

$$E(t) = a_0 + a_1 t + \frac{1}{2} a_2 t^2$$

The smoothing statistics are triple-exponential:

$$S_t^{(1)}(X) = \alpha[X(t)] + \beta[S_{t-1}^{(1)}(X)]$$

$$S_t^{(2)}(X) = \alpha[S_t^{(1)}(X)] + \beta[S_{t-1}^{(2)}(X)]$$

$$S_t^{(3)}(X) = \alpha[S_t^{(2)}(X)] + \beta[S_{t-1}^{(3)}(X)]$$

Initial conditions for the statistics are:

$$S_0^{(1)}(X) = a_0(0) - \left[\frac{\beta}{\alpha}\right]a_1(0) + \left[\frac{\beta(2-\alpha)}{4\alpha^2}\right]a_2(0)$$

$$S_0^{(2)}(X) = a_0(0) - \left[\frac{2\beta}{\alpha}\right]a_1(0) + \left[\frac{2\beta(3-2\alpha)}{4\alpha^2}\right]a_2(0)$$

$$S_0^{(3)}(X) = a_0(0) - \left[\frac{3\beta}{\alpha}\right]a_1(0) + \left[\frac{3\beta(4-3\alpha)}{4\alpha^2}\right]a_2(0)$$

A forecast made at time, T, extending periods into the future with respect to time, T, is given by:

$$\hat{X}_t(T) = \hat{a}_0(T) + \hat{a}_1(T)\tau + \frac{1}{2}\hat{a}_2(T)\tau^2$$

where,

$$\hat{a}_0(T) = 3S_t^{(1)}(X) - 3S_t^{(2)}(X) + 3S_t^{(3)}(X)$$

$$\hat{a}_1(T) = \frac{\alpha}{2\beta}\left[(6-5\alpha)S_t^{(1)}(X) - 2(5-4\alpha)S_t^{(2)}(X) - (4-3\alpha)S_t^{(3)}(X)\right]$$

$$\hat{a}_2(T) = \frac{\alpha^2}{\beta^2}\left[S_t^{(1)}(X) - 2S_t^{(2)}(X) + S_t^{(3)}(X)\right]$$

The primary advantages of using higher-level smoothing functions are twofold:

1. Given that the behavior of process can, in fact, be approximated by a mathematical model, the higher-order terms will describe higher-order (presumably decreasingly significant) effects, and result in a more accurate forecast, especially for times relatively "distant" in the future.

2. The response time of the model to changes in the process being monitored, as reflected in the co-efficients of the model, is shortened.

Some disadvantages of using excessively high-order functions are:

1. The increased storage and numerical analysis attendant to generating a forecast.

2. A tendency toward unstable transients in the model when the monitored process changes sharply. This effect is usually noticed when the co-efficient of the model vary significantly from those which theoretically describe the process. In attempting to correct itself, a high-order model will respond fast, but does so at the expense of a series of over-corrections.

For the purposes of this study, it was felt that the condition of constantly-changing trends was sufficiently sophisticated to serve as a "first-approximation" model.

One problem which was encountered with all referenced exponential smoothing methods is that they presume that data is sampled at regular intervals, causing the regular geometric decay of past data with time. However, if data is not so sampled, but rather, received at irregular intervals, then the geometric decay itself occurs irregularly, with a consequent introduction of randomness of sampling not assumed in the original equations, resulting in a commensurate loss of accuracy. In order to achieve a decay of past data proportionate to its age, the standard smoothing constant, α, is modified to a "smoothing function" of time-lag between the last reception of data and the "present" reception. In order to maintain an exponential decay with time, the smoothing function, α^*, will be defined as follows:

$$\alpha^* = \alpha^{\left[\frac{\Delta t}{t_b}\right]}$$

in which,

α is a fixed constant between 0 and 1 inclusive, and represents the unmodified, or "pure" smoothing constant, t_b is a time unit, (which will be defined for purposes of argument as the "time-base"), in multiples of which the actual time-lag may be specified;

Δt is the time-lag between reception of current data and the last reception of data.

A decrease in α, a decrease in t_b, or an increase in Δt, will all serve to decrease the significance of past data. In an obvious manner,

$$\beta^* = 1 - \alpha^*$$

and α^* and β^* are substituted for α and β as appropriate in the above equations. There is a slight flaw in the above approximation from a rigorous standpoint. The mathematics involved in the derivation of the smoothing relationships presume that α and β are constant. The above approximation makes them time-dependent. It is possible to propose other decay relationships to allow for the continuous arrival pattern of data. For the purposes of this paper, and the sample implementation, a rigorous and exact solution was not employed. A distinct reduction of average predictive error in sample data resulted from the use of this approximation, and it might be further reduced by a rigorous solution.

B. DISCUSSION OF GRAPHS

1. Figure 2 is a plot of one commodity option, May (1970) Idaho Potatoes, traded on the Chicago Mercantile Exchange on January 27, 1970, as reported by the Exchange ticker. Plotting symbols are as follows:

(1) Offers to sell (ask) are indicated by "+";

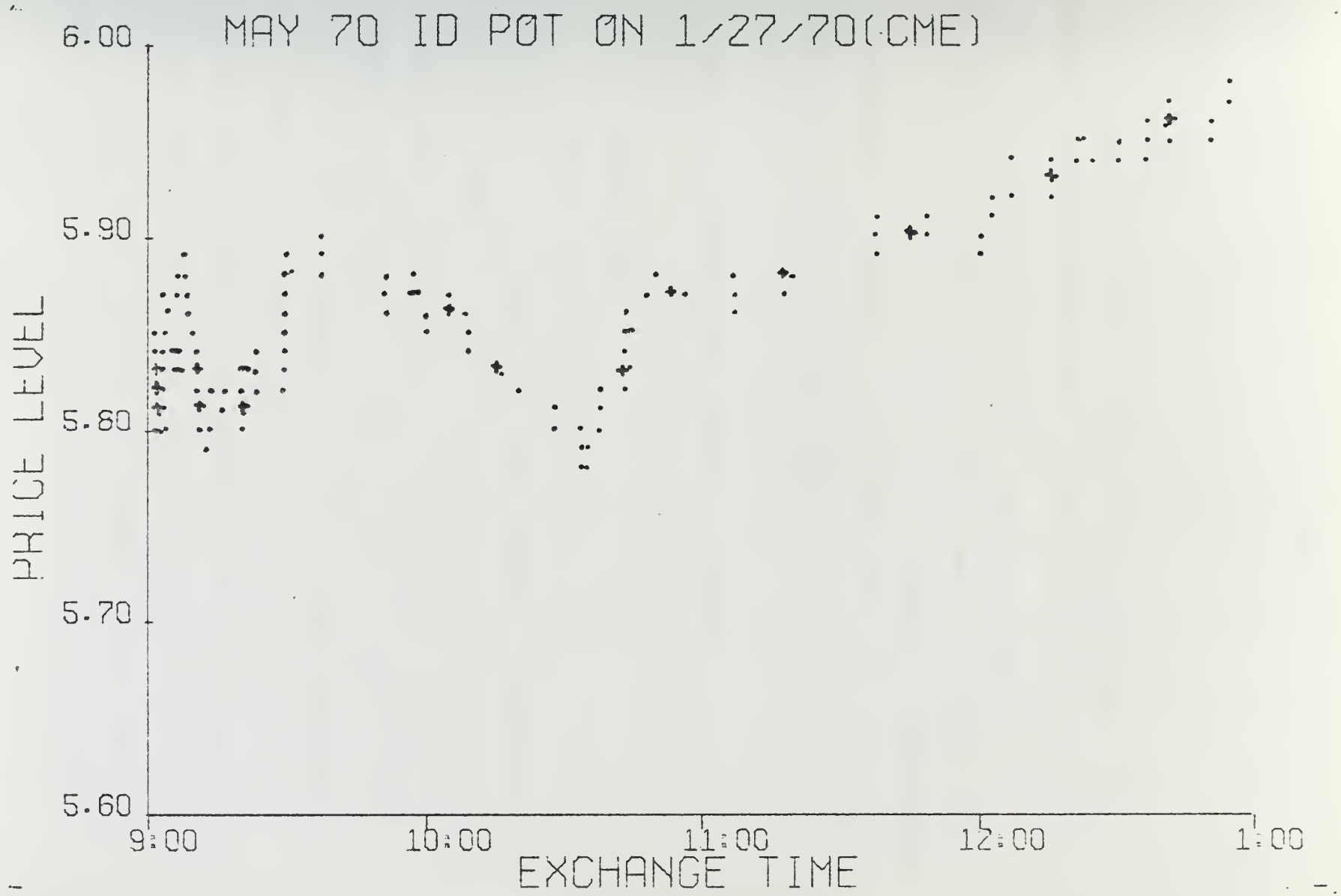
MAY 70 ID POT ON 1/27/70(CME)
6.00
5.90
5.80
5.70
5.60
PRICE LEVEL
9:00
10:00
11:00
12:00
1:00
EXCHANGE TIME

FIGURE 2

(2) Offers to buy (bid) are indicated by "-"; and,

(3) Completed transactions are reported by ".".

A sparcity of data in the 11:00 to 11:30 region results from missing sections of the ticker tape used as raw data.

2. Figure 3.

Figure 3 displays the effect of various time-bases upon the effective smoothing constant for a typical inter-data time lag, in this case, of approximately seven seconds. Sample time- bases of 1, 5, 10, 50, 100, and 500 seconds were used, and are indicated as such on Figure 3. It will be noted that the effective decay (SMOOTHING CONSTANT (MODF)) is significant for low values of the time-base over the entire range of the smoothing constant ("SMOOTHING CONSTANT (PURE"), whereas it is almost negligible in the 100-second and 500-second time-base region. This is expected from the exponential function of equation,

3. Figures 4 and 5

Figures 4 and 5 demonstrate the mean relative error for the days transactions in this commodity for various unmodified time-base-modified values of the smoothing constant. This mean relative error is determined by regenerating a forecast equation as each new EIU is "received" for this option, and determining the mean error of the remaining known data points. Obviously, the analysis is designed such that this advanced data is not used in making a prediction, only in analyzing the accuracy of that prediction. A mean error is recalculated from the "beginning of the day" for smoothing constant values from 0.01 through 0.99 inclusive, and the smoothing constant is, in fact, held constant through a days evaluation for that constant. Although spikes are noted at various points on

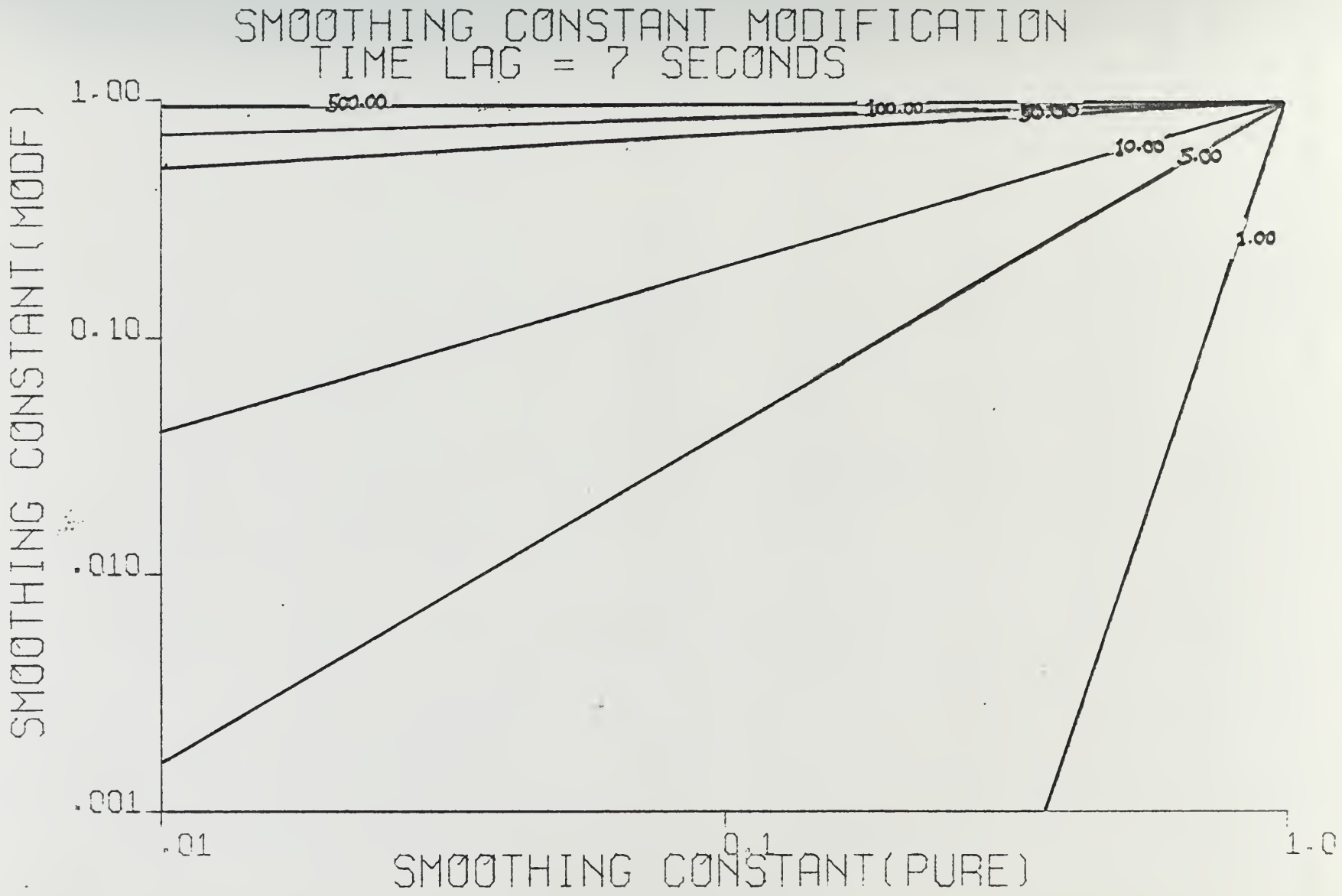
SMOOTHING CONSTANT MODIFICATION
TIME LAG = 7 SECONDS
SMOOTHING CONSTANT(MODF)
1.00
0.10
.010
.001
500.00
100.00
50.00
10.00
5.00
1.00
.01
0.1
1.0
SMOOTHING CONSTANT(PURE)

FIGURE 3

both graphs, this indicates an unavoidable data dependency, and in itself indicates only that exponential smoothing is an accurate technique under certain conditions. However, it will be noted "by eye" that a time-base-modified smoothing constant has a significantly lower mean error than an unmodified constant.

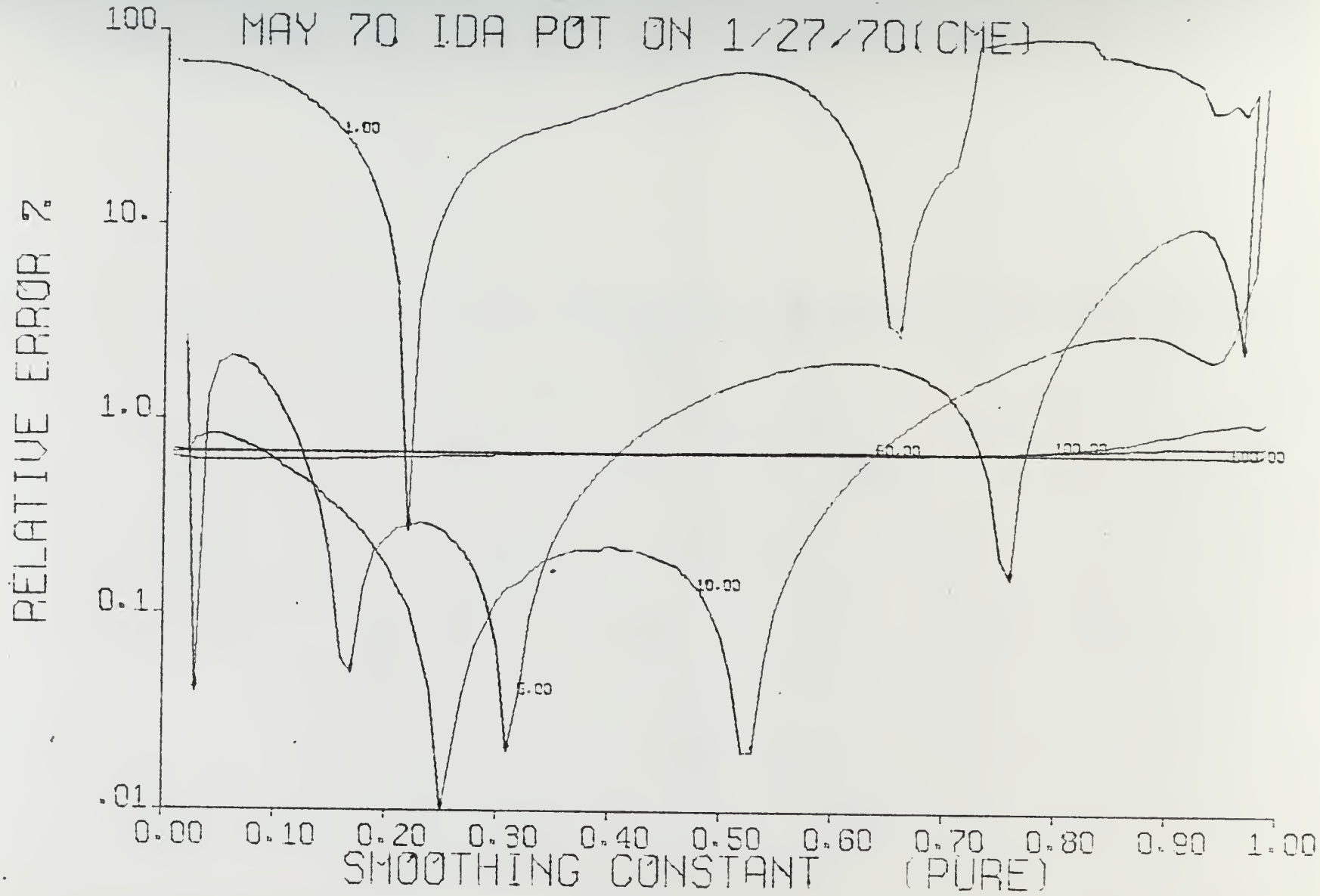
MAY 70 IDA POT ON 1/27/70(CME)
RELATIVE ERROR %
100
10.
1.0
0.1
.01
1.00
5.00
10.00
0.00
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00
SMOOTHING CONSTANT (PURE)

FIGURE 4

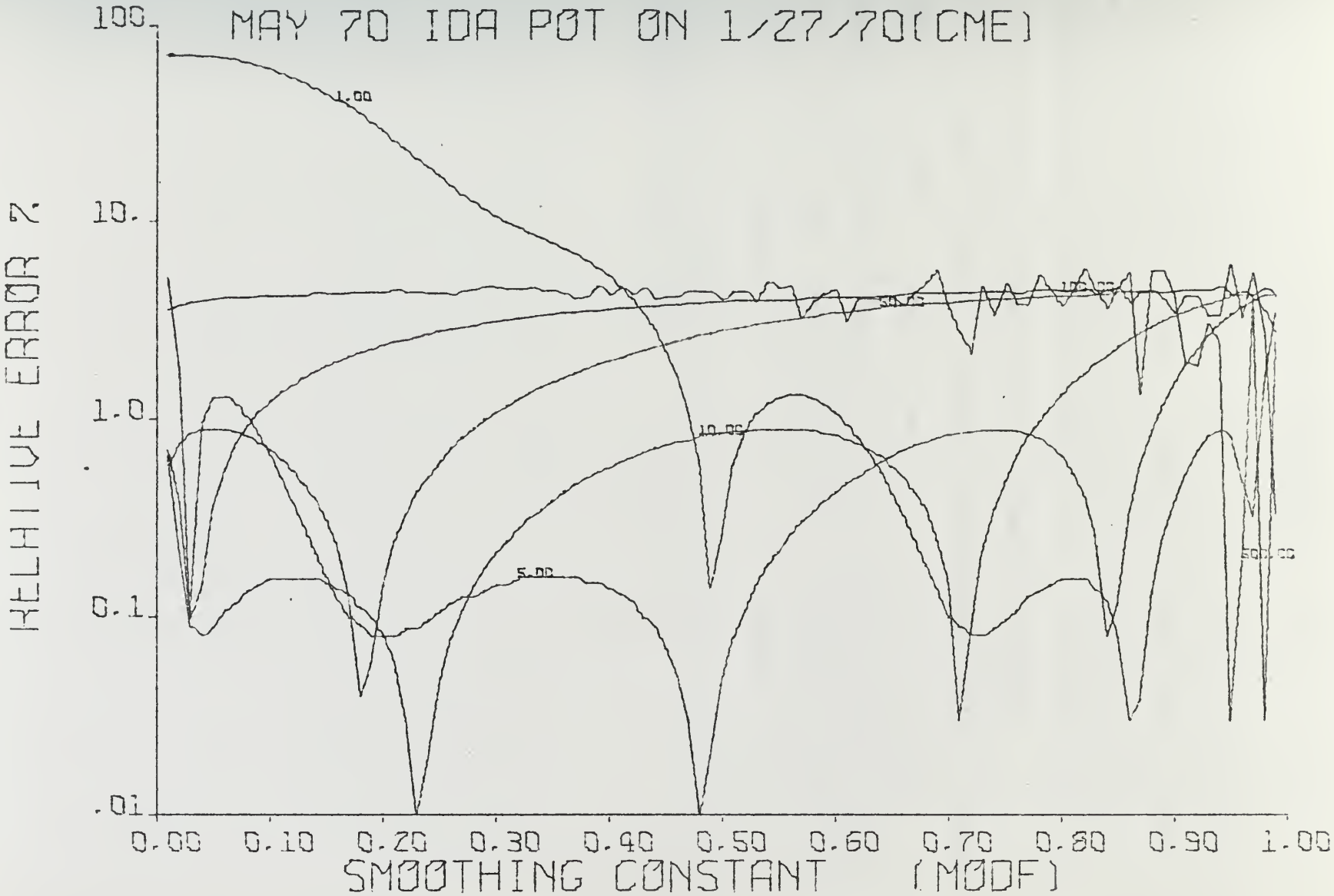
MAY 70 IDA POT ON 1/27/70(CME)
RELATIVE ERROR %
100.
10.
1.0
0.1
.01
1.00
5.00
10.00
0.00
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00
SMOOTHING CONSTANT (MODF)

FIGURE 5

APPENDIX C

TEXT PRE-PROCESSOR

As has been previously mentioned, an ideal text preprocessor might be one which optimized a trade-off between scan-length and maintenance of system state indicators. Work was abandoned on the implementation of a text editor as not contributing enough directly to merit the effort. It is, nevertheless, an integral part of a functional system.

To provide test data to simulate an exchange interface, a combined SNOBOL and FORTRAN job processed raw ticker text from the Chicago Mercantile Exchange, and converted it to pre-defined information units. In a real-time system, this approach would prove extremely inefficient, both from the standpoint of core and time. This program listing, as well as a half-hour segment of processed data for five commodities, is included for reference only.

PRE-PROCESSOR INPUT AND OUTPUT

A. TICKER TRANSMISSION OF CHICAGO MERCANTILE EXCHANGE, JANUARY 27, 1970

B. EXCHANGE INFORMATION UNITS (EIU) FOR 5 COMMODITY OPTIONS

CHICAGO MERCANTILE EXCHANGE, TRANSMISSION OF JAN. 27, 1970, 10:00 AM TO 10:30 AM

0.42.....LCQ31.15.....SALES.TO10AM.EG591EH376EJ42EK53EM9EU5.LBK1.LBN1.PK430PB
G567PBH326.PBK108PBN90PBQ25LCG113LCJ205.LCM79LCQ64LV*CV13LCZ12.LHG46LHJ21LHM
22.....PBG44.12..44.15..44.12...44.10..44.07..44.05...44.00..43.95..43.92
..43.90..43.87..43.90...43.92..43.95...PBH43.10..43.07..43.05...43.00..42.95.
.42.90..42.85..42.80..PBK42.95...42.92...A42.87..PBK42.85...B42.87PBK42.90..
42.87...42.85..42.80..42.77..A42.75PBK42.70B42.75PBK41..P*RPT.PBK42.80A42.77
PBK42.75..PBN42.42..42.40.....PBG42*3.9243.90..43.87..43.85..43.87.....PBH42
.82..42.85..42.80.....EG51.40..51.35..51.30..51.35...A51.30...EG51.25...EH4
6.00..46.10...A46.05.....EJ.....LBH78.00.....PK.B5.86PK5.87..A5.86.....PBG43.
85..43.87..43.90..43.92..43.95..43.92..43.90..43.87..43.87...43.85...
43.87...43.90...43.92..43.95...PBH42.75...42.80...42.85...42.80...42.75...42
.82..42.80..42.85..42.90..42.85..42.90...1005AM..PBK42.70B42.72PBK42.75..42.
72..42.70MAKE.LAST.PBK42.72READ.A42.72...PBK.B42.72B42.75PBK42.77..42.80...B
42.82...PBK42.85...PBN42.35..42.30..42.35..42.40..42.35..42.40...PBQ.A40.80P
BQ40.75..40.70..40.67..40.65.LCG29.40..LCJ130.40LCJ30...PPT.LCJ30...RPT.LCJ.
A30.40LCJ30.30...LCM.A31.12LCM31.10..LCQ31.12..31.15.....PK5.86.....EG51.3
0..51.35.....51.30.....EH46.00..46.05...46.00..469*.05..46.10EJ.A40.20EJ40.15.
40.10..40.20B40.25A40.25A40.20.....PK5.85...5.84.....LHG.A28.27LHG28.25..
28.30..28.25..B28.27LHG28.30.....PBG43.97..44.00..43.97...43.95..44.00...
43.95..43.97...44.00..44.02...44.05...44.02...44.05...44.07..SPREAD.PBG.&.PB
K120POINTS.....PBH42.95..42.97..43.00...43.02..43.05...43.00..43.05...43.00
42.97..43.00..43.02...43.00...1010AM.....CHGO.SPOT.BTR.MKT.CLOSED.I*UNQUOTED.
...CHGO.SPOT.EGGS.STDS.A51/12...CHGO.SPOT9*.EGGS.MKT.CLOSED.&.THERE.WERE.NO.
SALES.TODAY.....PBG.B44.10PBGB44.12PBG44.15..44.17..44.15..44.12...A44.10A44
.07PBG44.05...PBH43.02..43.05..43.07..43.05..43.00....PBK42.85..42.87..42.90.
.42.87...B42.90B42.92PBK42.95..A42.92A42.90A42.87A42.85B42.87....PBN42.35A42
.30B42.30B42.35PBN42.40..B42.42...PBN42.42..42.40.....PBQ40.65...40.67..B40.
80PBQ40.82A40.80PBQ40.75B40.80PBQ40.82..40.85...B40.90PBQ40.92.....LHJ.A27.07
LHJ27.05A27.02LHJ27.00..27.02.....MKT.NEWS.SURVEY.QUOTES..CHGO.SPOT.BTR.TONE
.STDY.CRY93SCR67/14..92SCR67/14..90SCR65/14EGGS.TONE.UNEVEN.WHT.LGE80PC.&.OV
ER.A59WHT.MDMS80PC.&.OVER.A54STDS80PC.&.OVER.B52OFF2.CKS37OFF1.....PK.A5.83P
K5.83.....EG51.35..51.30..B51.30..B51.40EG51.45..51.40..51.50...51.55...51.60
.....EH46.15..46.20..46.15...B46.20EH'6*:*46.25.....EJ.B40.20..1015AM.....EJ
40.20..40.15..B40.20.....P..EK38.10..37.90..B37.95B38.00B38.05B38.10.....
EU.A..PBG44.00B44.02PBG44.05..44.10..44.05..44.07..44.10...44.12..44.15...44
.17..44.20..44.15...A44.12..PBG.B44.12PBG44.15..44.17..44.20..44.15.....PBG4
4.20..44.17..PBH43.02..43.00..B43.05PBH43.07..43.10..43.12..43.15...43.12..2
22..RPT.PBH.A43.12.....PBH43.10..43.07..43.10..43.15...43.17..43.20..43.17...
PBK.B42.95B42.97PBK43.00..43.02..43.00B43.02PBK43.05A43.02B..PBK43.00B43.02.
1.60EG51.55B51.60EG51.65...51.70..A51.65EG51.60...B51.65EG51.70...51.75...B5
1.80EG51.85...1020AM.....EH46.20B46.25EH46.30..46.25..46.35..46.40...4
6.45..46.50.....EV.B37.50A37.70..EZ.B39.00A39.50.....OMAHA.LIVE.CATTLE.MKT.
TONE.SLAUGHTER.STEERS.RATHER.SLOW.STDY.HEIFERS.FAIRLY.ACTIVE.STDY.TO25HIGHER
.SALES.CHOICE.STEERS.1000TO1275LBS.GRADE2TO4..27.50TO28.50.....PBG44.20..44.
22..44.25..A44.22..PBG44.22..44.20..44.17..44.22..44.25..44.22...44.20..44.1
7..44.15....PBH43.20..43.17A43.15PBH43.12..43.10...43.12..43.15...43.17...43

```
.15...43.12...PBK43.05A43.02PBK43.00..B43.02..PBK43.05..43.02.....PBN.B42.50
BB42.52PBN42.55...42.50.......LCG29.45..29.47..29.50...A29.47.....LCJ30.45...3
0.42..30.40B30.42LCJ30.45.....30.42..30.40..LCM31.12..31.10...LCQ31.12..31.15.
....LCV30.60.......LHG28.25B28.27LHG28.30..28.27..28.30.....1025AM....LHJ27.05....
....PBG44.17..44.20..44.22..44.25..44.27..44.30..44.32..44.35..44.32..44.30..44.2
7..44.30.....PBH43.17..43.20..43.22..43.25...43.30..43.32...43.30..43.27.......
PBK43.05B43.07PBK43.10.....PBN.B42.52PBN42.55....42.60..SPREAD.PBN..&.PBG165POINTS
....PBQ.A40.90PBQ40.85......PBN42.55.......PK5.81..5.80.....EG51.80..51.75..A51.70E
G51.60..51.55..51.50..51.55.......EH.A46.45A46.40EH46.40...46.35...A46.30..EH.A
46.25EH46.20B46.25EH46.25B46.30EH46.30...46.25.....EJ.B40.25B40.30B40.40EJ40.4
5..40.50..40.45......EK.B38.15EK38.20......EM37.75.....EU40.15.......LBN87.00.
.....LCG.A29.45LCG29.42.....LCJ30.37..30.35...B30.37...LCW31.12..31.10.....LCQ3
1.12...LCV.B30.65.......PBG44.32..44.30..44.27..44.25....44.27..44.30..44.27..
44.30..44.32..44.30....PBN42..RPT.PBN....PBH43.30...43.25....43.27....A43.25....A4
3.22...PBK43.07..43.10.....1030AM..SALES.TO1030AM.EG775AM.EG*H460EJ61EK70EM9
```

CHICAGO MERCANTILE EXCHANGE, PRE-PROCESSED "INFORMATION UNITS"
JANUARY 27, 1970, 10:00 AM TO 10:30 AM, FOR TRANSAMISSION ACROSS
EXCHANGE INTERFACE

```
LCQ;  ;31.15;10;      0
 EG;V;0591.;10;     36
 PK;V;0430.;10;   1290
PBQ;V;0025.;10;   2330
LCQ;V;0064.;10;   3154
LHM;V;0022.;10;   4086
 EG;  ;51.40;10;  18566
 EG;  ;51.35;10;  18817
 EG;  ;51.30;10;  19068
 EG;  ;51.35;10;  19318
 EG;A;51.30;10;  19641
 EG;  ;51.25;10;  19999
 PK;B;05.86;10;  22114
 PK;  ;05.87;10;  22329
 PK;A;05.86;10;  22580
PBQ;A;40.80;10;  37094
PBQ;  ;40.75;10;  37418
PBQ;  ;40.70;10;  37702
PBQ;  ;40.67;10;  37986
PBQ;  ;40.65;10;  38270
LCQ;  ;31.12;10;  42040
LCQ;  ;31.15;10;  42324
 PK;  ;05.86;10;  42770
 EG;  ;51.30;10;  43337
 EG;  ;51.35;10;  43621
 EG;  ;51.30;10;  43945
 PK;  ;05.85;10;  48081
 PK;  ;05.84;10;  48364
PBQ;  ;40.65;10;  73962
PBQ;  ;40.67;10;  74243
PBQ;B;40.80;10;  74525
PBQ;  ;40.82;10;  74806
PBQ;A;40.80;10;  75017
PBQ;  ;40.75;10;  75298
PBQ;B;40.80;10;  75509
PBQ;  ;40.82;10;  75791
PBQ;  ;40.85;10;  76037
PBQ;B;40.90;10;  76354
PBQ;  ;40.92;10;  76635
 PK;A;05.83;10;  85111
 PK;  ;05.83;10;  85322
 EG;  ;51.35;10;  85709
 EG;  ;51.30;10;  85955
 EG;B;51.30;10;  86236
```

```
 EG;B;51.40;10;  86518
 EG;  ;51.45;10;  86764
 EG;  ;51.40;10;  87010
 EG;  ;51.50;10;  87256
 EG;  ;51.55;10;  87538
 EG;  ;51.60;10;  87819
 PK;  ;05.82;10;114562
 EG;  ;51.70;10;115135
 EG;A;51.65;10;115421
 EG;A;51.60;10;115707
 EG;  ;51.55;10;116041
 EG;B;51.60;10;116327
 EG;  ;51.65;10;116661
 EG;  ;51.70;10;117042
 EG;A;51.65;10;117424
 EG;  ;51.60;10;117758
 EG;B;51.65;10;118187
 EG;  ;51.70;10;118521
 EG;  ;51.75;10;118903
 EG;B;51.80;10;119332
 EG;  ;51.85;10;119666
LCQ;  ;31.12;10;147049
LCQ;  ;31.15;10;147340
PBQ;A;40.90;10;159607
PBQ;  ;40.85;10;159901
 PK;  ;05.81;10;160322
 PK;  ;05.80;10;161042
 EG;  ;51.80;10;161484
 EG;  ;51.75;10;161742
 EG;A;51.70;10;162036
 EG;  ;51.60;10;162294
 EG;  ;51.55;10;162552
 EG;  ;51.50;10;162809
 EG;  ;51.55;10;163067
LCQ;  ;31.12;10;173411
 EG;V;0775.;10;180056
```

TEST TEXT-PREPROCESSOR

SOURCE LISTING

```
//BOX107SL JOB (0445,0334FT,PMN9),'BENTLEY'
//STEPLIB DD DSN=SYS1.SNOBOL,DISP=SHR,VOL=SER=MVTREX,UNIT=2314
//PASS1 EXEC PGM=SNOBOL4,TIME=7,REGION=160K
//*
//*       THIS JOB STEP WILL CAUSE A FIRST-PASS EXTRACTION OF ALL
//*       DATA EXCEPT CALCULATION OF TRANSMISSION TIME.
//*
//*       DEFINE CARD READER
//*
//FT05F001 DD DDNAME=SYSIN
//*
//*       DEFINE SYSTEM PRINTER
//*
//FT06F001 DD SYSOUT=A
//*
//*       DEFINE SYSTEM CARD PUNCH
//*
//FT07F001 DD SYSOUT=B,SPACE=(TRK,(10,2))
//*
//*       DEFINE CHARACTER STREAM INPUT
//*
//FT10F001 DD DSN=S0445.CMETP1,UNIT=2314,VOL=SER=LINDA,
//    DISP=(OLD,KEEP),DCB=(LRECL=80,BLKSIZE=800,RECFM=FB)
//*
//*       DEFINE TEMPORARY STORAGE FOR RAW "INFORMATION UNITS"
//*
//GO.FT20F001 DD DSN=&RAWINF,DISP=(NEW,PASS),UNIT=SYSDA,
//    SPACE=(CYL,(181)),DCB=(LRECL=33,BLKSIZE=3300,RECFM=FB)
//*
//*       LOAD MAINLINE PROGRAM
//*
//SYSIN DD *
-LIST LEFT
*
*
*          THE FOLLOWING IS A SNOBOL4 PROGRAM TO EXTRACT COMMODITY
*          TRANSACTION DATA FROM A CONTINUOUS STREAM OF CHARACTERS, AS
*          WOULD BE TRANSMITTED BY THE CHICAGO MERCANTILE EXCHANGE.
*
*          THIS PROGRAM WILL EXTRACT DATA FOR THE COMMODITY MNEMONICS
*          DEFINED "VALID.NAME" BELOW.  THIS PROGRAM WILL EXTRACT, AS
*          A RAW "INFORMATION UNIT", THE "FUTURE OPTION DESIGNATION", IN
*          STANDARD C.M.E. MNEMONIC FORM, AN ACTION QUALIFIER (ASK,BID,
*          SALE), AND A NUMERICAL VALUE, AS A TWO-DECIMAL PLACE DECIMAL
*          FRACTION, AND THE LOCATION OF THE ACTION IN A CHARACTER STRING
*          BETWEEN TWO SUCCESSIVE EXCHANGE TIME STAMPS.  THIS INFORMATION
*          IS WRITTEN ONTO A DISK FILE FOR A SECOND-STAGE PROCESSING
*          BY A FORTRAN-IV PROGRAM WHICH CALCULATES, BY INTERPOLATION,
```

```
*          THE "EXACT" TIME THAT THE INFORMATION UNIT WAS TRANSMITTED.
*
*          THIS PROGRAM DOES NOT EXTRACT VOLUME REPORTS, ALTHOUGH THIS
*          WOULD PROVE A TRIVIAL ADDITION.
*
*-----> INITIALLIZE SYSTEM
*
*                DEFINE INPUT/OUTPUT DEVICES
*
  INPUT('TICKER',10,75)
 OUTPUT('PUNCH',20,'(33A1)')
*
*                DEFINE PATTERNS
*
 DIGIT = '0123456789'
 VALID.HOUR = '8' | '9' | '10' | '11' | '12' | '1'
 VALID.NAME = 'EG' | 'PK' | 'PBQ' | 'LHM' | 'LCQ'
 VALID.TRANS = (ANY('AB.') | '')   SPAN(DIGIT) '.' ANY(DIGIT)
.    ANY(DIGIT)
*
*                INITIALLIZE VARIABLES
*
 HOUR = '8'
 MINUTES = '27'
 INSTR = ''
*
*-----> GET A NEW INTER-TIME STRING
*
*                SET TIME STAMP AT BEGINNING OF STRING
*
NEW.PASS OLD.MIN = MINUTES
 OLD.HOUR = HOUR
*
*                CONTINUE BUILDING THE INPUT STRING UNTIL A VALID
*                TIME STAMP IS ENCOUNTERED.
*
READ.LINE INSTR = INSTR TICKER :F(WRAPUP)
 INSTR ((SPAN(DIGIT) 'AM.') . TMST) =  '(' :S(BRKOFF)
 INSTR ((SPAN(DIGIT) 'PM.') . TMST) =  '(' :F(READ.LINE)
*
*                BREAK OFF NEW INTER-TIME-STAMP STRING
*
BRKOFF INSTR (BREAK('(') . OPSTR) = ''
 INSTR '(' = ''
*
*                EXTRACT TERMINAL TIME STAMP OF STRING
*
 TMST SPAN(DIGIT) . TIME
```

```
 MINUTES = '00'
 LE(SIZE(TIME),2) :S(JUST.HRS)
 TIME ((LEN(2) RPOS(0)) . MINUTES) = ''
JUST.HRS TIME (VALID.HOUR RPOS(0)) . HOUR
*
*-----> PICK OUT AND WRITE VALID TRANSACTIONS
*
*
*                 CALCULATE OVERALL LENGTH OF INITIAL STRING
*
 FULL.STRING = SIZE(OPSTR)
*
*                 HUNT THROUGH STRING UNTIL FIRST VALID COMMODITY
*                 MNEMONIC FOUND
*
*                 THE OCCURRANCE OF A NEW MNEMONIC IN THE STRING
*                 TERMINATES A STRING OF TRANSACTIONS RELATING TO THE
*                 LAST MNEMONIC FOUND.
*
*
FIND.NAME OPSTR (VALID.NAME . NAME) = '$' :F(NEW.PASS)
 OPSTR BREAK('$') = ''
 OPSTR '$' = ''
*
*                 "DATA.GROUP" CONTAINS ALL TRANSACTIONS PERTAINING
*                 TO LAST VALID MNEMONIC FOUND.
*
 OPSTR (SPAN(DIGIT 'AB.') . DATA.GROUP) = ''  :F(FIND.NAME)
 PARTIAL.OPSTR = SIZE(OPSTR)
*
*                 PICK OUT TRANSACTIONS AND QUALIFERS, SCANNING DOWN
*                 THE STRING.
*
FIND.TRANS DATA.GROUP (VALID.TRANS . TRANS) = '$' :F(FIND.NAME)
 DATA.GROUP BREAK('$') = ''
 DATA.GROUP '$' = ''
 QUAL = ' '
 TRANS ((POS(0) ANY('AB')) . QUAL) = '' :S(TEST.TRANS.LENGTH)
 TRANS (POS(0) ('.' | NAME)) = ''
TEST.TRANS.LENGTH LE(SIZE(TRANS),5) :S(GOOD.TRANS)
*
*                 IF THE TRANSACTION LENGTH IS GREATER THAN FIVE
*                 CHARACTERS, THIS IS CONSIDERED TO BE IN ERROR.
*
 OUTPUT = '----> EXTRACTION (' QUAL TRANS ') IGNORED BETWEEN '
.   OLD.HOUR ':' OLD.MIN ' AND ' HOUR ':' MINUTES
.   :(FIND.TRANS)
*
```

```
*                CALCULATE THE LOCATION OF THE TRANSACTION IN THE
*                ORIGINAL INTER-TIME-STAMP STRING, IN NUMBER OF
*                CHARACTERS FROM THE BEGINNING.
*
GOOD.TRANS LOCUS = FULL.STRING - PARTIAL.OPSTR - SIZE(DATA.GROUP) - 4
*
*                PAD ALL EXTRACTED DATA ELEMENTS WITH BLANKS, SO
*                THAT THEY WILL RESULT IN A STANDARD FORMAT OUTPUT
*                RECORD.
*
 EQ(SIZE(NAME),3) :S(TEST2)
 NAME = ' ' NAME
TEST2 EQ(SIZE(TRANS),5) :S(TEST3)
 TRANS = '0' TRANS
TEST3 GE(LOCUS,0) :S(TEST4)
 LOCUS = '0000' :(TEST5)
TEST4 EQ(SIZE(LOCUS),4) :S(TEST5)
 LOCUS = '0' LOCUS :(TEST4)
TEST5 EQ(SIZE(FULL.STRING),4) :S(TEST6)
 FULL.STRING = '0' FULL.STRING
TEST6 EQ(SIZE(OLD.HOUR),2) :S(TEST7)
 OLD.HOUR = '0' OLD.HOUR
TEST7 EQ(SIZE(HOUR),2) :S(DUMP)
 HOUR = '0' HOUR
DUMP    HOUR '01' = '13'
*
*                CONSTRUCT AND TRANSMIT OUTPUT RECORD TO FILE.
*
 OUTPUT.LINE =              NAME ';' QUAL ':' TRANS ';' LOCUS ';'
.    FULL.STRING ';' OLD.HOUR ':' OLD.MIN ';' HOUR ':' MINUTES
 PUNCH = OUTPUT.LINE
.    :(FIND.TRANS)
*
*-----> CLOSE DOWN SYSTEM
*
WRAPUP
END
//PASS2 EXEC FORTCLGP,
//    PARM.FORT='NAME=PASS2'
//*
//*       THIS PASS CALCULATES AN INTERPOLATED "EXACT" TIME OF
//*       TRANSMISSION FOR THE INFORMATION UNIT AND RESTRUCTURES
//*       THE FORMAT OF THE INFORMATION UNIT, FOR FINAL
//*       TRANSMISSION ACROSS SIMULATED INTERFACE
//*
//FORT.SYSIN DD *
C
C               THIS PROGRAM CALCULATES THE TIME OF TRANSMISSION
```

```
C          FROM DATA GENERATED BY "PAS1" SNOBOL-4 PROGRAM
CC
CC------> DEFINE LOCAL STORAGE
C
      INTEGER*4 DATA(3),LOC,LEN,OHRS,OMIN,NHRS,NMIN,SHRS,SHSEC
      INTEGER*4 START
C
CCC               TITLE OUTPUT LISTING
C
      WRITE(6,4000)
 4000 FORMAT('1',T30,'PRE-PROCESSED INPUT FOR OPERATING SYSTEM')
C
CCC               READ A RECORD FROM TEMPORARY FILE CREATED BY LAST STEP
CC
   10 READ(20,1000,END=900)(DATA(I),I=1,3),LOC,LEN,OHRS,OMIN,NHRS,NMIN
 1000 FORMAT(3A4,2(I4,1X),3(I2,1X),I2)
      IF(OMIN.EQ.75)OMIN=30
C
CCCC              SUBROUTINE TMSPN CALCULATES TIME IN HOURS AND HUNDREDTHS
CCCCC         OF A SECOND OF THE TRANSACTION INTO ITS INTER-TIME-STAMP
CCCC          STRING
CC
      CALL TMSPN(SHRS,SHSEC,OHRS,OMIN,NHRS,NMIN,IRTN)
      IF(IRTN.EQ.2)GOTO 10
C
CCC               CALCULATE HUNDREDTHS OF A SECOND INTO HOUR
C
      START = IFIX((FLOAT(LOC)/FLOAT(LEN))* FLOAT(SHSEC))+OMIN*6000
   90 IF(START.LT.360000)GOTO 100
      START = START - 360000
      OHRS = OHRS + 1
      GOTO 90
C
CCCC              TRANSMIT RE-FORMATTED RECORD TO OUTPUT FILE, WHICH
CCCCC             IS FINAL FORMAT FOR TRANSMISSION ACROSS EXCHANGE
CCCC              INTERFACE
C
  100 WRITE(7,2000)(DATA(I),I=1,3),OHRS,START
 2000 FORMAT(3A4,I2,':',I6)
C
CCC               TRANSMIT COPY OF FINAL RECORD TO PRINTER
C
      WRITE(6,3000)(DATA(I),I=1,3),OHRS,START
 3000 FORMAT(1X,3A4,I2,':',I6)
      GOTO 10
  900 CALL EXIT
      END
C
```

```
C
      SUBROUTINE TMSPN(HRS,HSEC,OLDHR,OLDMN,NEWHR,NEWMN,RTRN)
C
C          SUBROUTINE TMSPN CALCULATES THE DIFFERENCE, IN HOURS AND
C          HUNDREDTHS OF A SECOND, BETWEEN TWO SUCCESSIVE TIMES
C          GIVEN IN HOURS AND MINUTES
C
C
C-----> DEFINE LOCAL STORAGE
C
      INTEGER*4 HRS,HSEC,OLDHR,OLDMN,NEWHR,NEWMN,RTRN
      INTEGER*4 WOLDHR,WOLDMN,WNEWHR,WNEWMN
C
C-----> EXECUTE
C
      WOLDHR = OLDHR
      WOLDMN = OLDMN
      WNEWHR = NEWHR
      WNEWMN = NEWMN
      IF(NEWMN.GE.OLDMN)GOTO 100
      IF(NEWHR.GT.OLDHR)GOTO 50
      RTRN = 2
      RETURN
   50 WNEWHR = WNEWHR - 1
      WNEWMN = WNEWMN + 60
  100 HRS = WNEWHR - WOLDHR
      HSEC =(WNEWMN - WOLDMN) * 6000
      RTRN = 1
      RETURN
      END
//*
//*       DEFINE INPUT DATA SET FROM LAST PASS
//*
//GO.FT20F001 DD DSN=*.PASS1.FT20F001,DISP=(OLD,DELETE),UNIT=SYSDA,
//    DCB=(LRECL=33,BLKSIZE=3300,RECFM=FB)
//*
//*       OUTPUT DATA SETS ARE:
//*            SYSTEM PRINTER: FT06F001;
//*            SYSTEM CARD PUNCH: FT07F001.
//*
```

APPENDIX D

SAMPLE IMPLEMENTATIONS

A. SUBROUTINE DESCRIPTION

In the sample implementations, subroutines were written to perform certain data and file manipulations as will be described below. Emphasis was solely upon data summarization, x storage and display; no attempt was made to implement user-position-monitoring functions, and functions which purported to interpret data other than by exponentially smoothed forecasting.

1. Subroutine BIRTH

Subroutine BIRTH initiallized the graphic overlay files, graphic display device and other bookkeeping functions of minimal significance from a general standpoint. It basically effects a "coldstart" upon the system.

2. Subroutine MNITR

Subroutine MNITR served as an experimental User Interface. Its function was two-fold:

a. Accept user requests in the form of light-pen indications of desired information, reset system state and branch parameters as appropriate and return to mainline. Mainline will overlay suitable routines into core to process request.

b. Allow user to control Exchange Interface for test purposes of program, by accepting a request for 1, 10 or 100 exchange information units (EIU's) through 2250 programmed-function keyboard.

3. Subroutine RECVR

Subroutine RECVR serves as an experimental Exchange Interface.

It accepts EIU's in a standard format from pre-processor simulator file, determines whether option is being monitored, and sets system state and branch variables as appropriate.

4. Subroutine UPDTE

Subroutine UPDTE performs functions described in Priority Levels One, Three and Five as follows:

a. It updates suitable historical graphic overlay files,

b. It updates the "time" line displayed by the resident graphic data set.

c. It updates appropriate exponential statistics and calculates co-efficients of a quadratic forecast model.

d. It erases and regenerates appropriate predictive graphic overlays .

5. Subroutine HIST

Subroutine HIST processes a user request for a new historical graphic overlay. It is one of the functions of Priority Level Two.

6. Subroutine PRED

Subroutine PRED processes a user request for the predictive graphic overlay of a commodity option whose historical graphic overlay is already being displayed. It is also one of the functions of Priority Level Two.

7. Subroutine ALPHA

Subroutine ALPHA permits the user to change the predictive smoothing constant (See Appendix B) while the system is in use.

8. Subroutine TBASE

Subroutine TBASE permits the user to change the "time-base" (See Appendix B) while the system is in use.

Subroutines HIST, PRED, ALPHA and TBASE are designed to be overlayable when called. Once their function is completed, their storage may be released.

9. Subroutine DOWN

Subroutine DOWN effects an orderly "normal termination" of the system. It transfers direct-access graphic overlay files to sequential offline storage media. It rewinds raw EIU storage media. It stores most system parameters necessary to effect a warmstart. It should, but does not (because the warmstart feature was not implemented) store all common blocks except those established by the Graphic Subroutine Package (GSP) to point o certain core-resident tables used by GSP. A warmstart would be effected by reloading all common blocks and reinitiallizing the 2250 graphic display unit. Finally, the subroutine produced significant statistical data which will be used by subroutine BIRTH the next time the system is coldstarted.

B. DYNAMIC OVERLAY CONSIDERATIONS

The dynamic overlay effected by OS/MVT is as follows:*

CONTROL SECTION	LEVEL ONE	LEVEL TWO
"COMMON"	1. BIRTH	1. INIT1
HASH		2. INIT2
		3. INIT3
		4. INIT4
		5. INIT5
		6. INIT6
	2. PRED	
	3. HIST	
	4. ALPHA	
	5. TBASE	
	6. RECVR	
	UPDTE	
	MNITR	

The Control Section occupies core at all times, consisting of the

mainline, "COMMON", and a hash-code generating subroutine, "HASH". Only of the six groups of Level Two may occupy core at any one time, and a subroutine called from mainline at this level overlays any subroutines previously in that storage region. The first five subroutines of Level One are infrequently called and are logically independent. The sixth group of three subroutines, MNITR, UPDTE and RECVR are called every time a new simulated EIU is to be processed. From the standpoint of speed, these routines will be core-resident as a group a large percentage of the time.

Similarly, initiallization subroutines on Level Two are logically independent, other than that they must be called by BIRTH sequentially. Resort to overlay did not significantly affect the core, reducing the occupied region by about 20% (from 150K to 120K). This is due primarily to the operating structure of the Graphic Subroutine Package, which occupies a very large block of core permanently under OS/MVT. In a more specifically-designed environment the above overlay structure should reduce the core-storage requirement significantly without significantly affecting operating efficiency.

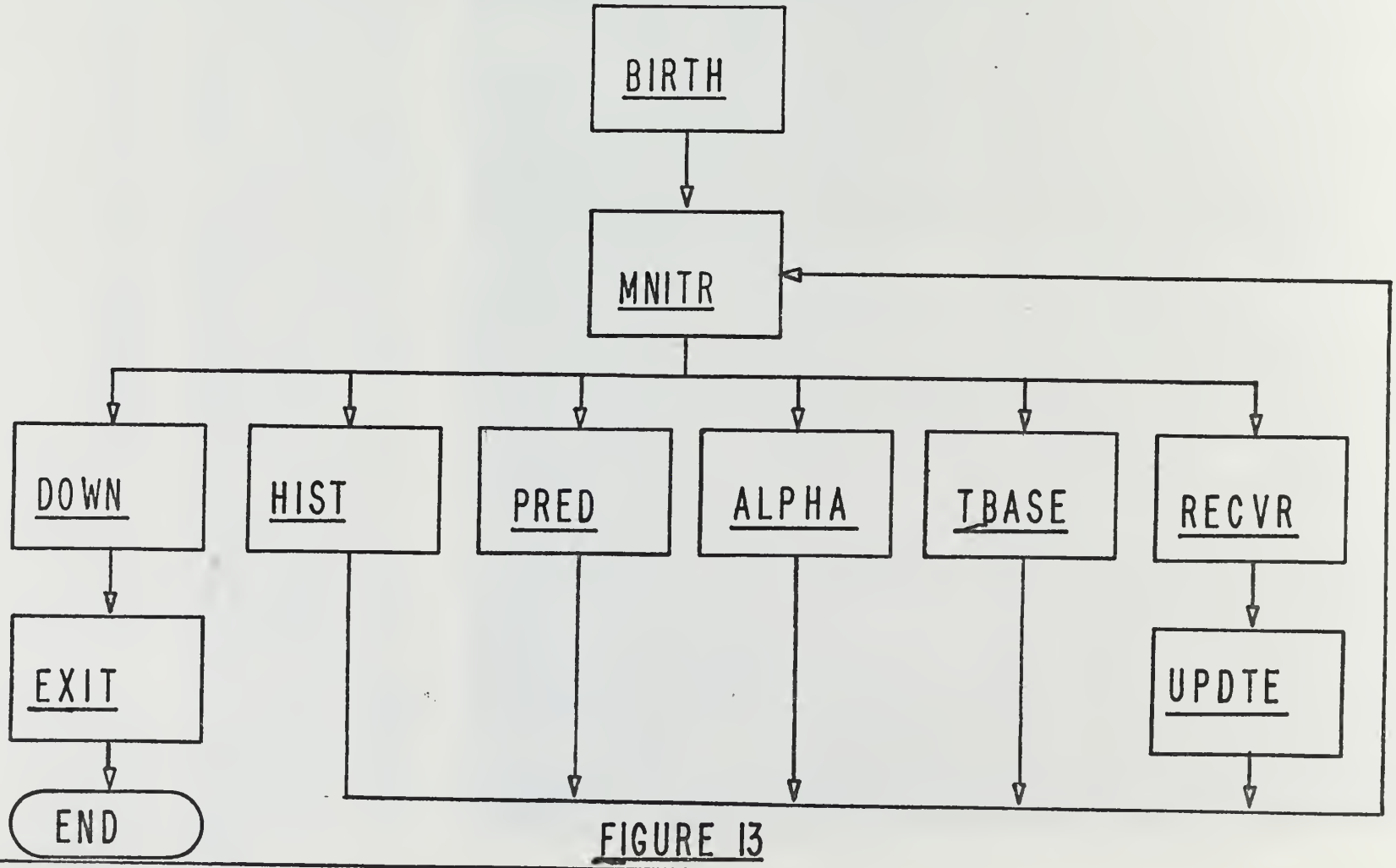

FIGURE 13

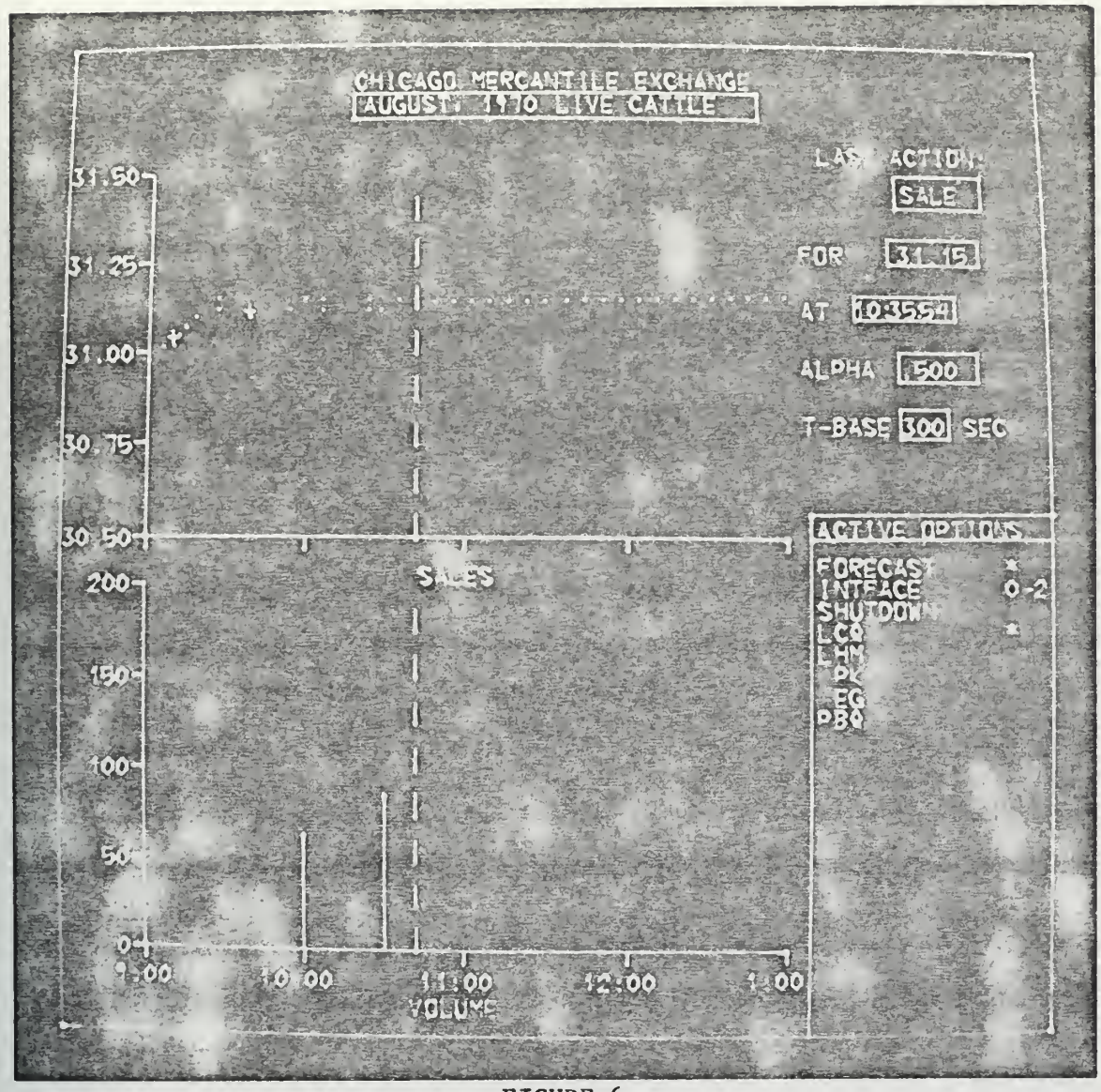

FIGURE 6

IBM-2250 display of August Live Cattle at about 10:40 a.m. Resident Display includes "LAST ACTION", "FOR", "AT", and "ACTIVE OPTIONS" table; also "SALES" and "VOLUME" axes and continuously updated "time" line. Historical overlay, paged into buffer by pointing light pen at one of mnemonics (e.g., "LCQ"), includes "SALES" points, "VOLUME" lines, vertical axis lables for "SALES" and "VOLUME", and "LAST ACTION", "FOR" and "AT" entries for option. Predictive overlay, paged into buffer by pointing light pen at "FORECAST", includes "ALPHA", "T-BASE" parameters, and forecast line.

Figure 7 shows complete day's transactions.

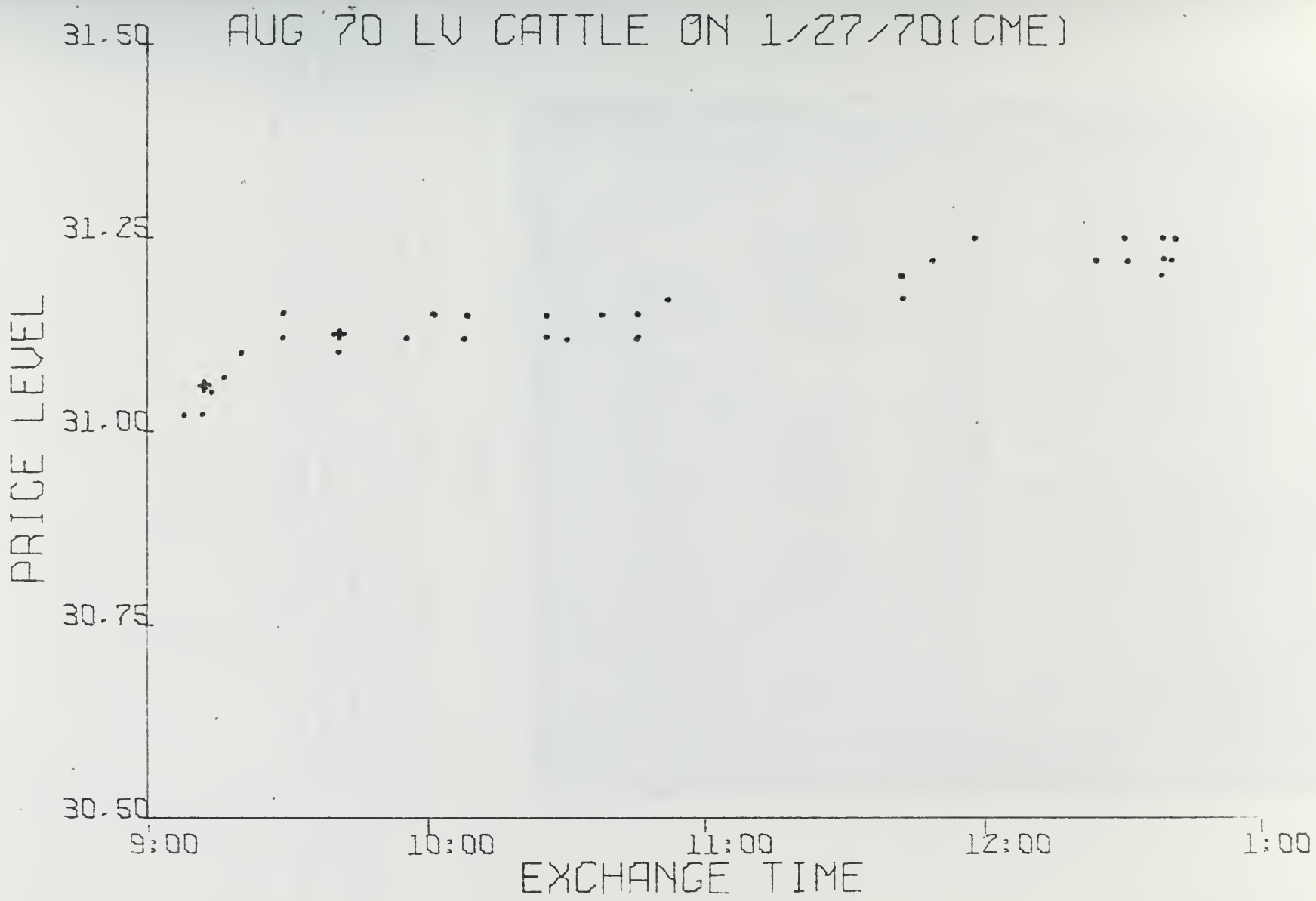
AUG 70 LV CATTLE ON 1/27/70(CME)
31.50
31.25
31.00
30.75
30.50
PRICE LEVEL
9:00
10:00
11:00
12:00
1:00
EXCHANGE TIME

FIGURE 7

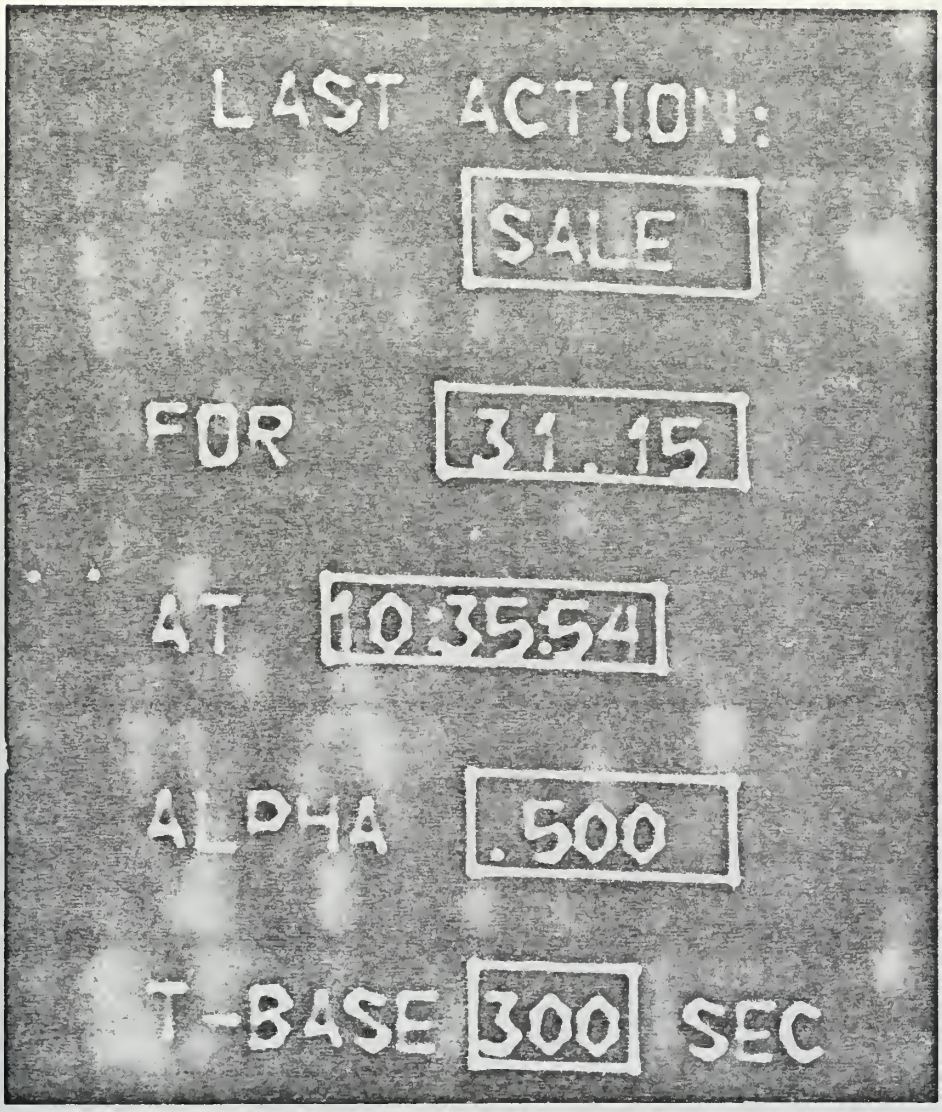

FIGURE 8

Closeup of displayed textual information for commodity option in figure 6. Historical information reveals that "LAST ACTION" was a "SALE" for 31.15 cents per pound, which occurred at 10:35:54 a.m. Predictive parametric information reveals that unmodified smoothing constant, "ALPHA" is 0.500 and that the time-base "T-BASE" is 300 seconds. All parameters and information in this frame are unique to specific options, and is paged into buffer when designated by user.

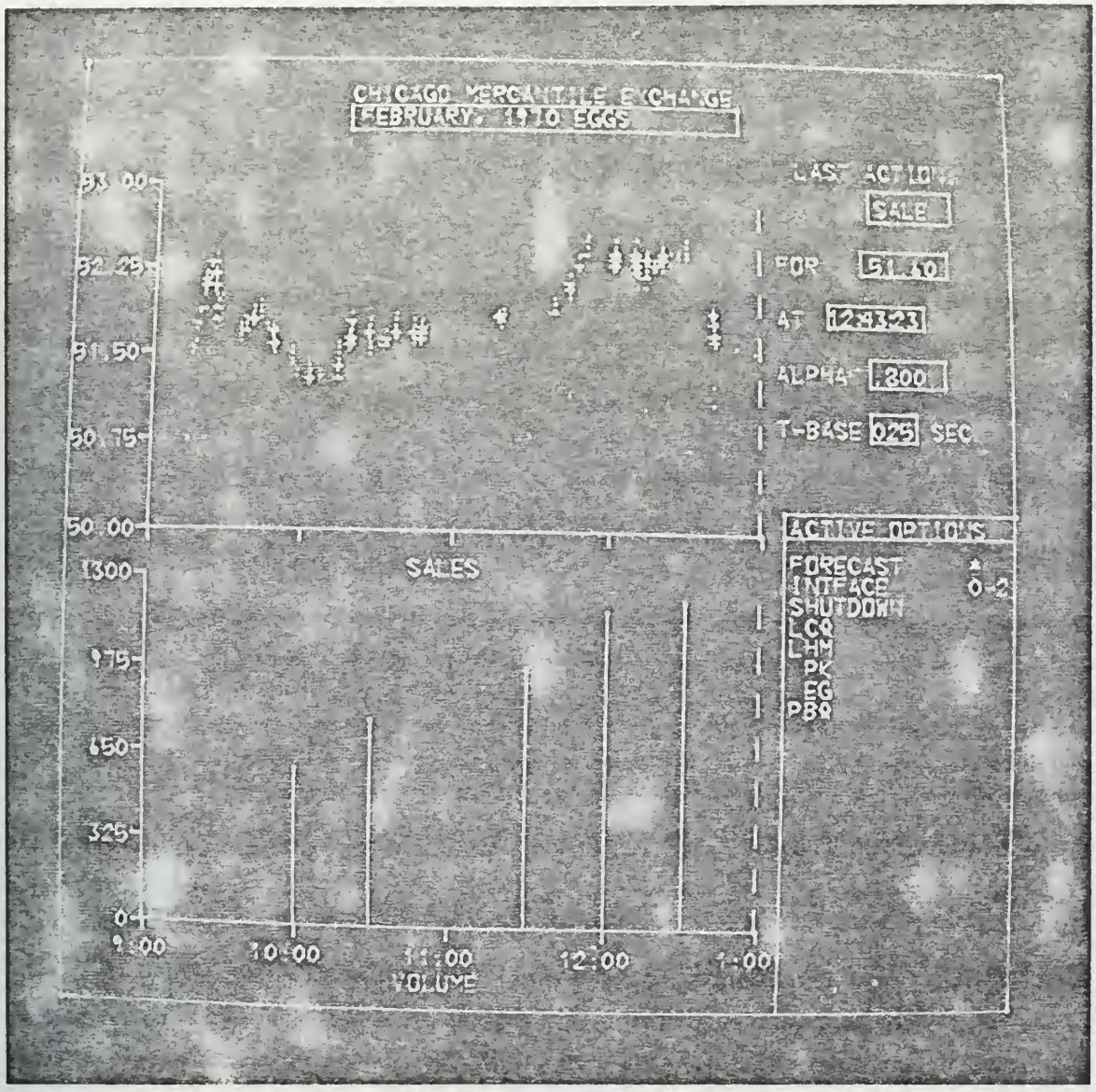

FIGURE 10

Display of all historical action for February Eggs (EG) for January 27, 1970 on Chicago Mercantile Exchange.

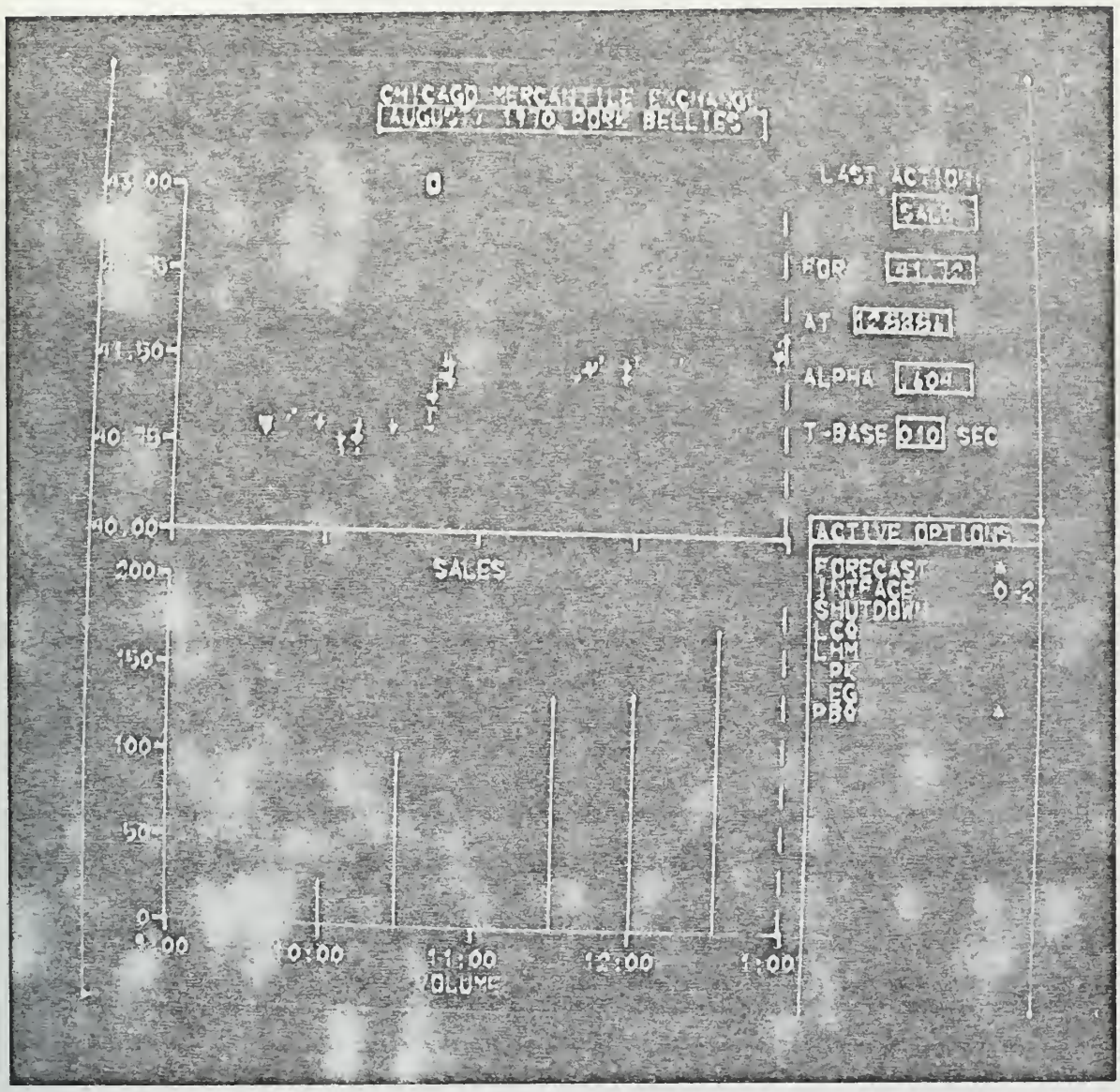

FIGURE 11

Display of all historical action for August Pork Bellies (PBQ) for January 27, 1970 on C.M.E. Note "0" at approximately 10:30 a.m., which indicates to user that "graphic overflow" has occurred at that time, because data point exceeded defined axis range. (A "U" will appear on horizontal axis in event of underflow). Repeated overflows or underflows would indicate need for re-definition of range. Missing data in 11:00 to 11:30 range results from incomplete ticker tape used for system test.

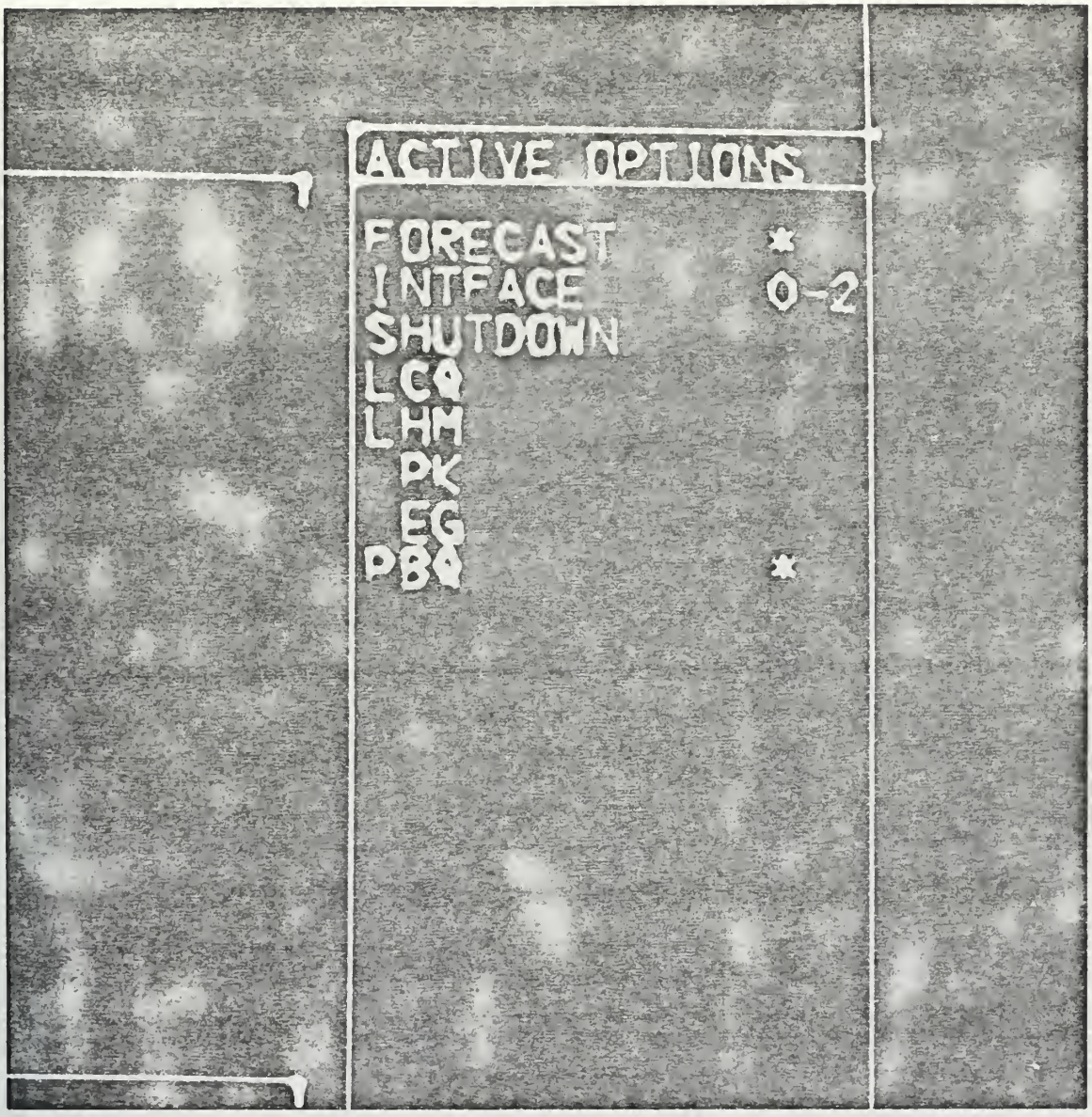

FIGURE 12

Enlarged view of "ACTIVE OPTIONS" table for Figure 11, shown at approximately actual size. "FORECAST" is turned on (indicated by asterisk) and off by touching word with 2250 light pen. "INTFCE" activates Exchange interface through 2250 programmed function box, transmitting 1, 10 or 100 test EIU's to system. Touching "SHUTDOWN" causes normal system termination. Touching "LCQ", "LHM", etc., causes historical overlay activated option to be paged into the buffer, and is indicated by asterisk. Subsequently touching "FORECAST" will activate and display forecast for this option.

COMMODITY MONITOR SYSTEM

SOURCE LISTING

```
//BOX107GT JOB (0445,0334FT,PMN9),'BENTLEY',TYPRUN=HOLD
//COMMON EXEC FORTCLGP,
//    TIME.FORT=1,
//    PARM.FORT='NAME=COMMON',
//    TIME.LINK=(,20),
//    PARM.LINK='MAP,LIST,OVLY',
//    TIME.GO=10,
//    REGION.GO=150K
//FORT.SYSPRINT DD SPACE=(CYL,(3,2))
//*
//*       DEFINE INPUT TO THE FORTRAN-G COMPILER
//*
//FORT.SYSIN DD *
C
C
C         COMMODITY MONITOR: MAINLINE CONTROL SECTION
C
C          THE MAIN CONTROL SECTION:
C
C               1. RESIDES PERMANENTLY IN CORE AS A CONTROL SECTION
C          FOR THE ENTIRE SYSTEM.  IT FACILITATES THE PASSAGE OF
C          CONTROL BETWEEN SUBROUTINES, AND THE DYNAMIC OVERLAY
C          OF ROUTINES WHERE APPROPRIATE.
C
C               2. ALLOCATE COMMON BLOCKS OF STORAGE FOR SUBROUTINES.
C
C               3. DEFINE DIRECT ACCESS FILES FOR SYSTEM
C
C               4. EFFECT MODIFICATIONS IN THE GRAPHIC SUBROUTINE
C          PACKAGE LINK/LOAD STATUS TO OPTIMIZE EFFICIENCY OF
C          EXECUTION.
C
C          REFERENCE IS MADE TO A LISTING OF SPECIFIC SUBROUTINES
C          FOR DETAILS OF A SPECIFIC SUBROUTINE'S FUNCTION.
C
C
C
C-----> DEFINE GLOBAL STORAGE
C              (FOLLOWING STATEMENTS WILL ESTABLISH FULL-WORD NAMED
C              COMMON BLOCKS OF SUBSCRIPTED SIZE.  INTERNAL BLOCK
C              LOCATIC ARE FURTHER DEFINED ONLY IF REFERENCED IN
C              THE CONTROL SECTION MAINLINE.)
      COMMON /DATE/ DATE(8)
C
C          DATE: CONTAINS CURRENT DATE
C
      COMMON /BUFF/ BUFF(5)
C
```

```
C          BUFF: TRANSFER BUFFER FOR HASH-ADDRESS GENERATOR
C
      COMMON /MSTB/ MSTB(4,19)
C
C        MSTB: MASTER TABLE, CONTAINING CURRENT MONITORED COMMODITIES
C
      COMMON /INPUT/ INPUT(5)
C
C        INPUT: INPUT BUFFER FOR EXCHANGE-INTERFACE
C
      COMMON /SYSSTE/ SYSSTE(6)
C
C        SYSSTE: SYSTEM STATE INDICATORS
C
      COMMON /SYSIO/ SYSIO(8)
C
C        SYSIO: SYSTEM INPUT/OUTPUT UNIT DESIGNATIONS
C
      INTEGER*4 BRNCH
      COMMON /SYSBRN/ BRNCH
C
C        SYSBRN: SYSTEM BRANCH INDICATOR
C
      COMMON /BLOCK/ BLOCK(527)
C
C        BLOCK: PRIMARY SYSTEM INPUT/OUTPUT BUFFERS
C
      INTEGER*4 INC40,INC50
      COMMON /DISKIO/ INC40,INC50,DISKIO(2)
C
C        DISKIO: SPECIFIES DIRECT-ACCESS I/O POINTERS AND GRAPHIC
C                DISPLAY BUFFER SIZES:
C
C          INC40: POINTS TO NEXT RECORD TO BE READ ON IO6
C          INC50: POINTS TO NEXT RECORD TO BE READ ON IO7
C
      INTEGER*4 VIDEO
      COMMON /GRAPH/ VIDEO,GRAPH(15)
C
C        GRAPH: CONTAINS PARAMETERS SPECIFIC TO GRAPHIC DISPLAY SYSTEM
C
C          VIDEO: GRAPHIC SUBROUTINE PACKAGE "GSPNAME"
C
C-----> DEFINE ACCESSED DIRECT ACCESS FILES
C
      DEFINE FILE 40(19,522,U,INC40)
      DEFINE FILE 50(19,74,U,INC50)
C
```

```
C          SET 360 OS ERROR MONITOR FOR FORTRAN TO IGNORE ANY LIMITS
C          ON NUMBER OF OVERFLOW OR UNDERFLOW ERRORS
C
      CALL ERRSET(208,1000,-1,1)
      CALL ERRSET(209,1000,-1,1)
    5 CALL BIRTH
C
C          LOAD ALL GRAPHIC SUBROUTINES THAT ARE REQUIRED FOR
C          SUBROUTINES MNITR,RECVR,UPDTE, PERMANENTLY.
C
      CALL SPEC(VIDEO,2,1,-56)
      CALL SPEC(VIDEO,1,13,17,26,-30,37,-41,51)
      CALL SPEC(VIDEO,1,53)
C
   10 CALL MNITR
      GOTO(100,200,300,400,500,600),BRNCH
C
  100 CALL RECVR
      GOTO(10,150),BRNCH
C
  150 CALL UPDTE
      GOTO 100
C
  200 CALL DOWN
  250 CALL EXIT
C
  300 CALL HIST
      GOTO 10
C
  400 CALL PRED
      GOTO 10
C
  500 CALL ALPHA
      GOTO 10
C
  600 CALL TBASE
      GOTO 10
C
      END
C


      SUBROUTINE BIRTH
C
C          SUBROUTINE BIRTH:
C
C               1. LOADS I/O DEVICE PARAMETERS;
C
```

```
C               2. LOADS DATE;
C
C               3. INITIALLIZES SYSTEM BY CALLING
C                    INITIALLIZATION SUBROUTINES.
C
C-----DEFINE GLOBAL STORAGE
C
      INTEGER*4 DATE
      COMMON /DATE/ DATE(8)
C
C         DATE: CONTAINS CURRENT DATE
C
      INTEGER*4 IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
      COMMON /SYSIO/ IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
C
C        SYSIO: SYSTEM INPUT/OUTPUT UNIT DESIGNATIONS
C
C           IO1: SYSTEM PRINTER
C           IO2: SYSTEM CARD READER
C           IO3: EXCHANGE INTERFACE INPUT FILE
C           IO4: GRAPHIC DISPLAY OUTPUT UNIT
C           IO5: RAW INFORMATION RECORDING UNIT (SEQUENTIAL)
C           IO6: HISTORICAL GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C           IO7: PREDICTIVE GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C           IO8: SYSTEM "SAVE" FILE WHEN SHUT DOWN (SEQUENTIAL)
C
C          (1) LOAD SYSIO,DATE,MODE
C
      READ(5,1000)IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8,MODE,(DATE(II),II=1,8)
 1000 FORMAT(8(I2,1X),I1,1X,8A1)
C
C          (2) CALL INTIALLIZATION SUBROUTINES
C
  100 CALL INIT1
      CALL INIT2
      CALL INIT3
      CALL INIT4
      CALL INIT5
      CALL INIT6
C
C          (3) RETURN
C
      RETURN
C
C          THIS IS A TRANSIENT ROUTINE WHICH MAY BE OVERLAID AFTER
C          EXECUTION HAS COMPLETED.
C
      END
```

```
C
C
      SUBROUTINE INIT1
C
C         SUBROUTINE INIT1:
C
C              1. CREATES THE "MASTER TABLE" IN CORE, WHICH CONTAINS
C         THOSE COMMODITIES BEING ACTIVELY MONITORED.   A COMMODITY
C         MNEMONIC IS LOCATED IN THE TABLE THROUGH A UNIQUELY-
C         GENERATED HASH ADDRESS USING THE MNEMONIC SYMBOLS THEMSELVES.
C         MNEMONICS WHICH COLLIDE WITH THOSE PREVIOSLY LOADED ARE
C         REJECTED, AND A SUITABLE ERROR MESSAGE IS PRINTED.
C
C              2. LOADS VARIOUS DATUM CONCERNING THE COMMODITY
C         DESCRIBED BY AN ACCEPTED MNEMONIC ONTO DIRECT-ACCESS STORAGE
C         FOR LATER USE BY THE SYSTEM, NAMELY: THE LITERAL TITLE
C         OF THE COMMODITY DESCRIBED BY THE MNEMONIC, AND DATA RELATING
C         TO ITS EVENTUAL HISTORIC AND PREDICTIVE GRAPHIC DISPLAYS.
C
C-----DECLARATION OF GLOBAL STORAGE
C
      INTEGER*4 MSTB1,MSTB2,MSTB3,MSTB4
      COMMON /MSTB/ MSTB1(19),MSTB2(19),MSTB3(19),MSTB4(19)
C
C        MSTB: MASTER TABLE, CONTAINING CURRENT MONITORED COMMODITIES
C
C          MSTB1: NUMBER OF SIGNIFICANT CHARACTERS IN MNEMONIC
C          MSTB2: THIRD CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED, IF ANY
C          MSTB3: SECOND CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED
C          MSTB4: FIRST CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED
C
      INTEGER*4 CHA,CHB,CHC,NOSIG,HSHCD
      COMMON /BUFF/ CHA,CHB,CHC,NOSIG,HSHCD
C
C          BUFF: TRANSFER BUFFER FOR HASH-ADDRESS GENERATOR
C
C          CHA: FIRST CHARACTER OF MNEMONIC, IF ANY, LEFT-JUSTIFIED
C          CHB: SECOND CHARACTER OF MNEMONIC, LEFT JUSTIFIED
C          CHC: THIRD CHARACTER OF MNEMONIC, LEFT-JUSTIFIED
C          NOSIG: NUMBER OF SIGNIFICANT CHARACTERS IN MNEMONIC
C          HSHCD: HASH ADDRESS GENERATED FROM MNEMONIC
C
      INTEGER*4 IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
      COMMON /SYSIO/ IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
C
C        SYSIO: SYSTEM INPUT/OUTPUT UNIT DESIGNATIONS
```

```
C
C          IO1: SYSTEM PRINTER
C          IO2: SYSTEM CARD READER
C          IO3: EXCHANGE INTERFACE INPUT FILE
C          IO4: GRAPHIC DISPLAY OUTPUT UNIT
C          IO5: RAW INFORMATION RECORDING UNIT (SEQUENTIAL)
C          IO6: HISTORICAL GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C          IO7: PREDICTIVE GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C          IO8: SYSTEM "SAVE" FILE WHEN SHUT DOWN (SEQUENTIAL)
C
      INTEGER*4 BLOCK,SUBBLK
      COMMON/BLOCK/BLOCK(522),SUBBLK(5)
C
C        BLOCK: PRIMARY SYSTEM INPUT/OUTPUT BUFFERS
C
C          BLOCK: SHARED HISTORICAL AND PREDICTIVE DIRECT-ACCESS
C          SUBBLK: RAW HISTORICAL RECORDER OUTPUT BUFFER
C
      INTEGER*4 INC40,INC50,MXBLK6,MXBLK7
      COMMON /DISKIO/ INC40,INC50,MXBLK6,MXBLK7
C
C        DISKIO: SPECIFIES DIRECT-ACCESS I/O POINTERS AND GRAPHIC
C                DISPLAY BUFFER SIZES:
C
C          INC40: POINTS TO NEXT RECORD TO BE READ ON IO6
C          INC50: POINTS TO NEXT RECORD TO BE READ ON IO7
C          MXBLK6: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO IO6 FILE
C          MXBLK7: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO IO7 FILE
C
C
C-----DEFINE LOCAL STORAGE
C
      INTEGER*4 TMBS,TMBST
      INTEGER*4 BLOCKS(74)
      INTEGER*4 PTR,TITLE(7)
      INTEGER*4 ENT,II
      REAL*4 ALPHA,ALPHAT
      REAL*4 LW1,HGH1,LW2,HGH2
      REAL*4 EXSM1(3)
C
C-----DEFINE RELEVANT EQUIVALENT STORAGE LOCATIONS
C
      EQUIVALENCE (BLOCK(1),BLOCKS(1))
      EQUIVALENCE(BLOCK(1),TMBS)
      EQUIVALENCE(BLOCK(2),ALPHA)
      EQUIVALENCE (BLOCK(7),LW1)
      EQUIVALENCE (BLOCK(8),HGH1)
      EQUIVALENCE (BLOCK(9),LW2)
```

```
      EQUIVALENCE (BLOCK(10),HGH2)
      EQUIVALENCE (BLOCK(11),TITLE(1))
      EQUIVALENCE (BLOCK(18),EXSM1(1))
C
C-----DEFINE ACCESSED DISK FILES
C
      DEFINE FILE 40(19,522,U,INC40)
      DEFINE FILE 50(19,74,U,INC50)
C
CC    (1) ZERO MSTB1
C
      DO 100 II = 1,19,1
  100 MSTB1(II) = 0
      LW2 = 0.0
C
CCC       (2) ZERO ENTRIES ON I06 AND I07
CCCC
CCCC          (A) ZERO ENTIRE BLOCK
C
      DO 110 II = 1,522
  110 BLOCK(II) = 0
C
CCC           (B) WRITE BLOCK ON ALL 19 SECTORS OF I06,I07
C
      DO 120 ENT = 1,19
      WRITE(I06'ENT)BLOCK
  120 WRITE(I07'ENT)BLOCKS
C
CC    (3) LOAD PATTERNS INTO TABLE FROM INPUT FILE.
CCC
CCCC      A. READ A CARD WITH RIGHT-JUSTIFIED PATTERN IN COL 1-3, AND
CCCC         NUMBER OF SIGNIFICANT CHARACTERS IN COL. 5. TEST "NOSIG".
C
  150 READ(I02,1000,END=700)CHA,CHB,CHC,NOSIG,(TITLE(JJ),JJ=1,7),LW1,HGH
     11,HGH2
 1000 FORMAT(3A1,1X,I1,1X,7A4,2(1X,F5.2),1X,F5.0)
      READ(I02,1001)(EXSM1(II),II=1,3),ALPHAT,TMBST
 1001 FORMAT(3(E9.2,1X),F5.3,1X,I3)
C
      IF(NOSIG)700,700,151
  151 IF(2 - NOSIG)160,170,160
  160 IF(3 - NOSIG)165,170,165
  165 WRITE(I01,3000)CHA,CHB,CHC,NOSIG
 3000 FORMAT(' INVALID INPUT. "',3A1,'" HAS ',I2,' SIGNIFICANT FIGURES.
     1   CARD SKIPPED.')
      GO TO 150
C
C         B. CALCULATE HASH CODE FOR TABLE THROUGH SUBROUTINE "HASH".
```

```
C
  170 CALL HASH
C
C          C. EXAMINE MSTB1(HSHCD). IF NOT ZERO, THEN COLLISION HAS
C             OCCURRED.  OTHERWISE OK TO INSERT ENTRY.
C
      IF(MSTB1(HSHCD))200,300,200
C
C          D. COLLISION OCCURRED, PRINT MESSAGE AND RETURN TO READ.
C
  200 WRITE(IO1,2000)CHA,CHB,CHC,HSHCD,MSTB2(HSHCD),MSTB3(HSHCD),MSTB4(H
     1SHCD)
 2000 FORMAT(' "',3A1,'" COLLIDES AT  ',I2,' WITH "',3A1,'" ALREADY THER
     1E. REJECTED.')
      GO TO 150
C
C          E. NO COLLISION. LOAD INTO TABLE.
C             WRITE OTHER DATA TEMPORARILY ON IO7
C
  300 MSTB2(HSHCD) = CHA
      MSTB3(HSHCD) = CHB
      MSTB4(HSHCD) = CHC
      MSTB1(HSHCD) = NOSIG
      TMBS = TMBST
      ALPHA = ALPHAT
      WRITE(IO7'HSHCD) BLOCKS
      GOTO 150
C
C     RETURN TO MAINLINE
C
  700 RETURN
C
C          THIS IS A TRANSIENT ROUTINE WHICH MAY BE OVERLAID AFTER
C          EXECUTION HAS COMPLETED.
C
      END
C
C


      SUBROUTINE HASH
C
C          SUBROUTINE HASH CALCULATES A HASH-ADDRESS FROM THE COMMODITY
C          MNEMONIC BY THE FOLLOWING METHOD:
C
C               1. ADD TOGETHER INTEGER REPRESENTATIONS OF ALL
C          SIGNIFICANT CHARACTERS IN MNEMONIC;
C
```

```
C              2. PERFORM MODULO-19-PLUS-1 OPERATION ON RESULTANT
CC          VALUE  TO GENERATE HASH-ADDRESS BETWEEN 1 AND 19.
CC
C-----DECLARATION OF GLOBAL STORAGE
C
      INTEGER*4 CHA,CHB,CHC,NOSIG,HSHCD
      COMMON /BUFF/ CHA,CHB,CHC,NOSIG,HSHCD
CCC
CCC         BUFF: TRANSFER BUFFER FOR HASH-ADDRESS GENERATOR
CCC
CCCCC        CHA: FIRST CHARACTER OF MNEMONIC, IF ANY, LEFT-JUSTIFIED
CCCCC        CHB: SECOND CHARACTER OF MNEMONIC, LEFT JUSTIFIED
CCCCC        CHC: THIRD CHARACTER OF MNEMONIC, LEFT-JUSTIFIED
CCCCC        NOSIG: NUMBER OF SIGNIFICANT CHARACTERS IN MNEMONIC
CCCCC        HSHCD: HASH ADDRESS GENERATED FROM MNEMONIC
CCCCC
C-----DECLARATION OF LOCAL STORAGE
C
      LOGICAL*1 CBT(4),NUBT(4)
      INTEGER*4 CHAR,NUCHR
C
C-----DEFINE RELEVANT EQUIVALENT STORAGE LOCATIONS
C
      EQUIVALENCE  (CHAR,CBT(1)),(NUCHR,NUBT(1))
C
CC    (1) CALCULATE SUM OF SECOND TWO CHARACTERS, THUS FOLDING THEM
C
      CHAR = CHB
      NUCHR = 0
      NUBT(4) = CBT(1)
      IHASH = NUCHR
      CHAR = CHC
      NUBT(4) = CBT(4)
      IHASH = IHASH + NUCHR
C
CC    (2) IF NCHAR EQUALS 3 FOLDING FIRST CHARACTER BEFORE MODULO-19
C
      GO TO(900,400,300),NOSIG
  300 CHAR = CHA
      NUBT(4) = CBT(4)
      IHASH = IHASH + NUCHR
C
CCC   (7) PERFORM MODULO-19 OPERATION
C
  400 HSHCD = IHASH - (IHASH/19)*19 + 1
C
CC    (4) RETURN TO MAINLINE
```

```
C
  900 RETURN
C
CC          THIS ROUTINE SHOULD BE PERMANENTLY CORE-RESIDENT FOR
C           MAXIMUM STEADY-STATE SPEED.
      END
C
CC


      SUBROUTINE INIT2
C
CC          SUBROUTINE INIT2 INITIALLIZES SYSTEM STATES AND BRANCH
CC
C-----DEFINITION OF GLOBAL STORAGE.
C
      INTEGER*4 STAT1,STAT2,STAT3,       STAT5,STAT6,STAT7
      COMMON /SYSSTE/ STAT1,STAT2,STAT3,       STAT5,STAT6,STAT7
C
CC        SYSSTE: SYSTEM STATE INDICATORS
CC
CC          STAT1:INDICATES NUMBER OF TIMES LEFT TO RECEIVE DATA (TEST)
CC          STAT2: INDICATES PREDICTIVE STATE ACTIVE IF EQUAL TO 2
CC          STAT3: INDICATES TYPE OF TRANSACTION ON EXCHANGE INTERFACE
CC             =2: ASK;
CC             =3: BID
CC             =4: SALE
CC             =5: VOLUME REPORT
CC          STAT5: INDICATES MASTER TABLE INDEX OF ACTIVE HISTORICAL OVERLAY
CC          STAT6: INDICATES WHETHER EXCHANGE INTERFACE OPEN OR CLOSED
CC          STAT7: MASTER TABLE INDEX OF ACTIVE PREDICTIVE DISPLAY
C
      INTEGER*4 BRN1
      COMMON /SYSBRN/ BRN1
C
CC        SYSBRN: SYSTEM BRANCH INDICATOR
CC
CC          THIS INDICATOR IS SET BY SUBROUTINES TO SHIFT SYSTEM CONTROL
CC          AFTER EXECUTION.
CC
CC
CC          (1) SET SYSTEM STATES
C
      STAT1 = 1
      STAT2 = 1
      STAT3 = 1
      STAT5 = 20
      STAT6 = 1
```

```
      STAT7 = 20
C
C         (2) SET SYSTEM BRANCH STATES
      BRN1 = 1
C
C         (3) RETURN TO MAINLINE
C
  700 RETURN
C
C         THIS IS A TRANSIENT ROUTINE WHICH MAY BE OVERLAID AFTER
C         EXECUTION HAS COMPLETED.
C
      END
C
C


      SUBROUTINE INIT3
C
C         SUBROUTINE INIT3:
C
C              1. INITIALLIZES GRAPHIC DISPLAY PARAMETERS, CORE
C         REGIONS AND DEVICE (IBM 2250-1):
C
C                   2. DEFINE DEVICE DEPENDENT DISPLAY PARAMETERS
C
C-----DEFINITION OF GLOBAL STORAGE
C
      INTEGER*4 IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
      COMMON /SYSIO/ IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
C
C        SYSIO: SYSTEM INPUT/OUTPUT UNIT DESIGNATIONS
C
C          IO1: SYSTEM PRINTER
C          IO2: SYSTEM CARD READER
C          IO3: EXCHANGE INTERFACE INPUT FILE
C          IO4: GRAPHIC DISPLAY OUTPUT UNIT
C          IO5: RAW INFORMATION RECORDING UNIT (SEQUENTIAL)
C          IO6: HISTORICAL GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C          IO7: PREDICTIVE GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C          IO8: SYSTEM "SAVE" FILE WHEN SHUT DOWN (SEQUENTIAL)
C
      INTEGER*4 VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      INTEGER*4 KEY,ATTN
      REAL*4 TO,DXDT,SLO,VLO
      COMMON /GRAPH/ VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      COMMON /GRAPH/ ATTN,KEY(5),TO,DXDT,SLO,VLO
C
```

```
C         GRAPH: CONTAINS PARAMETERS SPECIFIC TO GRAPHIC DISPLAY SYSTEM
C
C           VIDEO: GRAPHIC SUBROUTINE PACKAGE "GSPNAME"
C           NULL: GRAPHIC SUBROUTINE PACKAGE "NULL" VARIABLE
C           I2250: GRAPHIC SUBROUTINE PACKAGE DEVICE NAME
C           IGDS1: RESIDENT GRAPHIC DATA SET ADDRESS
C           IGDS2: HISTORICAL OVERLAY GRAPHIC DATASET ADDRESS
C           IGDS3: PREDICTIVE OVERLAY GRAPHIC DATA SET ADDRESS
C           ATTN: PRIMARY ATTENTION LEVEL BASE ADDRESS
C           KEY (ALL): RELATIVE ADDRESSES OF SPECIFIEC GRAPHIC ORDERS
C                      IN IGDS2 HISTORICAL FILE
C           TO: SCREEN LOCATION OF 9:00 A.M.
C           DXDT: INCHES PER HOUR ON DISPLAY SCREEN
C           SLO: VERTICAL ORIGIN FOR LOWER LIMIT OF SALES ON SCREEN
C           VLO: VERTICAL ORIGIN FOR "0" FOR VOLUME ON SCREEN
C
      INTEGER*4 MXBLK6,MXBLK7
      COMMON /DISKIO/ DISKIO(2),MXBLK6,MXBLK7
C        DISKIO: SPECIFIES DIRECT-ACCESS I/O POINTERS AND GRAPHIC
C                DISPLAY BUFFER SIZES:
C
C           INC40: POINTS TO NEXT RECORD TO BE READ ON I06
C           INC50: POINTS TO NEXT RECORD TO BE READ ON I07
C           MXBLK6: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I06 FILE
C           MXBLK7: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I07 FILE
C
C
C-----> DEFINE LOCAL STORAGE
C
      INTEGER*4 MXBLK
C
C -----> INITIALLIZE VARIABLES
C
C           RESIDENT GRAPHIC DISPLAY WILL HAVE 6 GDOA UNITS
C
      MXBLK = 256 * 6
C
C           HISTORICAL GRAPHIC OVERLAYS WILL HAVE 8 GDOA UNITS
C
      MXBLK6 = 256 * 8
C
C           PREDICTIVE GRAPHIC OVERLAY WILL HAVE 1 GDOA UNIT.
C
      MXBLK7 = 256 * 1
C
      NULL = 10
C
C           (1) INITIALLIZE GSP PACKAGE
```

```
C
      CALL INGSP(VIDEO,NULL)
      CALL INDEV(VIDEO,IO4,I2250)
      CALL TMDEV(I2250)
      CALL TMGSP(VIDEO)
      CALL INGSP(VIDEO,NULL)
C
CC         (2) INITIALLIZE DEVICE
C
      CALL INDEV(VIDEO,IO4,I2250)
C
CC         (4) INITIALLIZE ALL GRAPHIC DATA SETS
CC
CC             IGDS1:  ELEMENTS COMMON TO ALL DISPLAYS.
CC             IGDS2:  ELEMENTS, HISTORICAL, OF ONE PATTERN.
CC             IGDS3:  ELEMENTS, PREDICTIVE, OF ONE PATTERN.
C
      CALL INGDS(I2250,IGDS1,MXBLK,1)
      CALL INGDS(I2250,IGDS2,MXBLK6,2)
      CALL INGDS(I2250,IGDS3,MXBLK7,1)
C
CC         (5) SET ALL DATA MODES TO REAL, ABSOLUTE
C
      CALL SDATM(IGDS1,1,1)
      CALL SDATM(IGDS2,1,1)
      CALL SDATM(IGDS3,1,1)
C
CC         (6) SET ALL GRAPHIC MODES TO OPTIMIZED
C
      CALL SGRAM(IGDS1,1)
      CALL SGRAM(IGDS2,2)
      CALL SGRAM(IGDS3,1)
C
CC         (7) SET ALL CHARACTER MODES TO BASIC, PROTECTED
C
      CALL SCHAM(IGDS1,3)
      CALL SCHAM(IGDS2,1)
      CALL SCHAM(IGDS3,1)
C
CC         (8) SET ALL GRAPHIC DATA SET LIMITS TO COVER ENTIRE SCREEN)
C
      CALL SGDSL(IGDS1,0.0,0.0,1.0,1.0,0.0,0.0,1.0,1.0)
      CALL SGDSL(IGDS2,0.0,0.0,1.0,1.0,0.0,0.0,1.0,1.0)
      CALL SGDSL(IGDS3,0.0,0.0,1.0,1.0,0.0,0.0,1.0,1.0)
C
CC         (9) SET ALL GRAPHIC DATA SET LIMITS TO:
CC
C              X:(0.0 -> 12.0)INCHES; Y:(0.0 -> 12.0) INCHES
```

```
C
      CALL SDATL(IGDS1,0.0,0.0,12.0,12.0)
      CALL SDATL(IGDS2,0.0,0.0,12.0,12.0)
      CALL SDATL(IGDS3,0.0,0.0,12.0,12.0)
C
CCC       (10) SET SCISSORING TO DATA SET BOUNDARIES, AND CONTINUE
CCC            GENERATION AFTER SCISSORRING
C
      CALL SSCIS(IGDS1,2)
      CALL SSCIS(IGDS2,2)
      CALL SSCIS(IGDS3,2)
C
CCC       (11) SET BEAM TO ZERO IN EACH DATA SET
C
      CALL STPOS(IGDS1,0.0,0.0)
      CALL STPOS(IGDS2,0.0,0.0)
      CALL STPOS(IGDS3,0.0,0.0)
C
CCC      (12) SET PHYSICAL SCREEN DEFINITION PARAMETERS
C
      TO = 1.5000
      DXDT = 7.000 / 4.0
      SLO = 6.0000
      VLO = 1.5000
C
C-----INITIALLIZATION FINISHED,RETURN TO MAINLINE
C
      RETURN
C
CCC       THIS IS A TRANSIENT ROUTINE WHICH MAY BE OVERLAID AFTER
CCC       EXECUTION HAS COMPLETED.
C
      END
CC
C
C
      SUBROUTINE INIT4
C
CCC       SUBROUTINE INIT4 CREATES THE RESIDENT GRAPHIC DISPLAY,
CCC       IGDS1
C
C-----DEFINITION OF GLOBAL STORAGE.
C
      INTEGER*4 VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      INTEGER*4 KEY,ATTN
      REAL*4 TO,DXDT,SLO,VLO
      COMMON /GRAPH/ VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
```

```
      COMMON /GRAPH/ ATTN,KEY(5),TO,DXDT,SLO,VLO
C
C        GRAPH: CONTAINS PARAMETERS SPECIFIC TO GRAPHIC DISPLAY SYSTEM
C
C          VIDEO: GRAPHIC SUBROUTINE PACKAGE "GSPNAME"
C          NULL: GRAPHIC SUBROUTINE PACKAGE "NULL" VARIABLE
C          I2250: GRAPHIC SUBROUTINE PACKAGE DEVICE NAME
C          IGDS1: RESIDENT GRAPHIC DATA SET ADDRESS
C          IGDS2: HISTORICAL OVERLAY GRAPHIC DATASET ADDRESS
C          IGDS3: PREDICTIVE OVERLAY GRAPHIC DATA SET ADDRESS
C          ATTN: PRIMARY ATTENTION LEVEL BASE ADDRESS
C          KEY (ALL): RELATIVE ADDRESSES OF SPECIFIEC GRAPHIC ORDERS
C                     IN IGDS2 HISTORICAL FILE
C          TO: SCREEN LOCATION OF 9:00 A.M.
C          DXDT: INCHES PER HOUR ON DISPLAY SCREEN
C          SLO: VERTICAL ORIGIN FOR LOWER LIMIT OF SALES ON SCREEN
C          VLO: VERTICAL ORIGIN FOR "0" FOR VOLUME ON SCREEN
C
C-----DEFINE LOCAL STORAGE
C
      INTEGER*4 II,ICRVL,IKEY
      INTEGER*4 TMKY
      REAL*4 XLOC,YLOC,YLOC1
C
C               A. DRAW BORDER
C
      CALL STPOS(IGDS1,0.5,0.5)
      CALL PLINE(IGDS1,0.5,11.5)
      CALL PLINE(IGDS1,11.5,11.5)
      CALL PLINE(IGDS1,11.5,0.5)
      CALL PLINE(IGDS1,0.5,0.5)
C
C               B. DRAW TITLE
C
      CALL PTEXT(IGDS1,'CHICAGO MERCANTILE EXCHANGE',27,NULL,NULL,NULL,
     13.85,11.1)
C
C               C. DRAW TITLE BOX UNDER MAIN TITLE
C
      CALL STPOS(IGDS1,3.75,10.62)
      CALL PLINE(IGDS1,3.75,10.93)
      CALL PLINE(IGDS1,8.25,10.93)
      CALL PLINE(IGDS1,8.25,10.62)
      CALL PLINE(IGDS1,3.75,10.62)
C
C               D. DRAW UPPER SET OF AXES AND TICS
C
```

```
      CALL STPOS(IGDS1,1.5,10.0)
      CALL PLINE(IGDS1,1.5,6.0)
      CALL PLINE(IGDS1,8.5,6.0)
      XLOC = 8.5
      DO 50 II = 1,5,1
      CALL PSGMT(IGDS1,XLOC,6.0,XLOC,5.875)
   50 XLOC = XLOC - 1.75
      YLOC = 10.0
      DO 100 II = 1,5,1
      CALL PSGMT(IGDS1,1.5,YLOC,1.375,YLOC)
  100 YLOC=YLOC-1.0
C
CCC               E. DRAW LOWER SET OF AXES AND TICS
C
      CALL STPOS(IGDS1,1.5,5.5)
      CALL PLINE(IGDS1,1.5,1.5)
      CALL PLINE(IGDS1,8.5,1.5)
      XLOC = 8.5
      DO 150 II = 1,5,1
      CALL PSGMT(IGDS1,XLOC,1.5,XLOC,1.375)
  150 XLOC = XLOC - 1.75
      YLOC = 5.5
      DO 200 II = 1,5,1
      CALL PSGMT(IGDS1,1.5,YLOC,1.375,YLOC)
  200 YLOC = YLOC - 1.0
C
CCC               F. TITLE UPPER AXES WITH "SALES"
C
      CALL PTEXT(IGDS1,'SALES',5,NULL,NULL,NULL,4.6,5.625)
C
CCC               G. TITLE LOWER AXES WITH "VOLUME" AND TIMES
C
      CALL PTEXT(IGDS1,'VOLUME',6,NULL,NULL,NULL,4.50,0.875)
      CALL PTEXT(IGDS1,'9:00',4,NULL,NULL,NULL,1.225,1.19)
      CALL PTEXT(IGDS1,'10:00',5,NULL,NULL,NULL,2.85,1.19)
      CALL PTEXT(IGDS1,'11:00',5,NULL,NULL,NULL,4.60,1.19)
      CALL PTEXT(IGDS1,'12:00',5,NULL,NULL,NULL,6.35,1.19)
      CALL PTEXT(IGDS1,'1:00',4,NULL,NULL,NULL,8.125,1.19)
C
CCC               H. WRITE "LAST ACTION" BOXES AND TITLES
C
      CALL PTEXT(IGDS1,'LAST ACTION:',12,NULL,NULL,NULL,9.0,10.25)
      CALL STPOS(IGDS1,9.75,9.625)
      CALL PLINE(IGDS1,9.75,10.00)
      CALL PLINE(IGDS1,10.75,10.00)
      CALL PLINE(IGDS1,10.75,9.625)
      CALL PLINE(IGDS1,9.75,9.625)
      CALL PTEXT(IGDS1,'FOR',3,NULL,NULL,NULL,8.75,9.10)
```

```
      CALL STPOS(IGDS1,9.65,9.00)
      CALL PLINE(IGDS1,9.65,9.275)
      CALL PLINE(IGDS1,10.70,9.275)
      CALL PLINE(IGDS1,10.70,9.00)
      CALL PLINE(IGDS1,9.65,9.00)
      CALL PTEXT(IGDS1,'AT',2,NULL,NULL,NULL,8.75,8.475)
      CALL STPOS(IGDS1,9.25,8.375)
      CALL PLINE(IGDS1,9.25,8.650)
      CALL PLINE(IGDS1,10.40,8.650)
      CALL PLINE(IGDS1,10.40,8.375)
      CALL PLINE(IGDS1,9.25,8.375)
      CALL PTEXT(IGDS1,':',1,NULL,NULL,NULL,9.640,8.525)
      CALL PTEXT(IGDS1,':',1,NULL,NULL,NULL,10.018,8.525)
C
CCC               I. GENERATE TIME LINE AT ZERO
C
      YLOC = 1.5
      ICRVL = 20
      DO 300 II = 1,8,1
      ICRVL = ICRVL + 1
      YLOC1=YLOC+0.25
      CALL PSGMT(IGDS1,1.5,YLOC,1.5,YLOC1,ICRVL)
  300 YLOC = YLOC + 0.50
      YLOC = 6.0
      DO 350 II = 1,8,1
      ICRVL = ICRVL + 1
      YLOC1 = YLOC + 0.25
      CALL PSGMT(IGDS1,1.5,YLOC,1.5,YLOC1,ICRVL,IKEY)
  350 YLOC = YLOC + 0.50
C
CCC               (J) GENERATE BOXES, TITLES AND DUMMY ENTRIES FOR PREDICTIO
C
      CALL BGSEQ(IGDS1,39)
      CALL PTEXT(IGDS1,'ALPHA',5,40,NULL,NULL,8.750,7.850)
      CALL STPOS(IGDS1,9.750,7.700)
      CALL PLINE(IGDS1,9.750,8.025)
      CALL PLINE(IGDS1,10.65,8.025)
      CALL PLINE(IGDS1,10.65,7.700)
      CALL PLINE(IGDS1,9.750,7.700)
      CALL PTEXT(IGDS1,'.500',4,41,NULL,NULL,9.850,7.850)
      CALL PTEXT(IGDS1,'T-BASE',6,42,NULL,NULL,8.750,7.225)
      CALL STPOS(IGDS1,9.750,7.075)
      CALL PLINE(IGDS1,9.750,7.400)
      CALL PLINE(IGDS1,10.30,7.400)
      CALL PLINE(IGDS1,10.30,7.075)
      CALL PLINE(IGDS1,9.750,7.075)
      CALL PTEXT(IGDS1,'SEC',3,NULL,NULL,NULL,10.55,7.225)
      CALL PTEXT(IGDS1,'300',3,43,NULL,NULL,9.850,7.225)
```

```
      CALL ENSEQ(IGDS1)
      RETURN
C
CC        THIS IS A TRANSIENT ROUTINE WHICH MAY BE OVERLAID AFTER
CC        EXECUTION HAS COMPLETED.
C
      END
C
C


      SUBROUTINE INIT5
C
CC        SUBROUTINE INIT5:
CC
CC             1. CREATES ATTENTION INTERRUPTS FOR THE GRAPHIC
CC        DISPLAY DEVICE (IBM 2250-1);
CC
CC             2. CREATES OPTIONS-TABLE IN RESIDENT GRAPHIC DISPLAY.
CC
C-----DEFINE GLOBAL STORAGE
C
      INTEGER*4 MSTB1,MSTB2,MSTB3,MSTB4
      COMMON /MSTB/ MSTB1(19),MSTB2(19),MSTB3(19),MSTB4(19)
C
CC        MSTB: MASTER TABLE, CONTAINING CURRENT MONITORED COMMODITIES
CC
CC          MSTB1: NUMBER OF SIGNIFICANT CHARACTERS IN MNEMONIC
CC          MSTB2: THIRD CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED, IF ANY
CC          MSTB3: SECOND CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED
CC          MSTB4: FIRST CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED
C
      INTEGER*4 VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      INTEGER*4 KEY,ATTN
      REAL*4 TO,DXDT,SLO,VLO
      COMMON /GRAPH/ VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      COMMON /GRAPH/ ATTN,KEY(5),TO,DXDT,SLO,VLO
C
CC        GRAPH: CONTAINS PARAMETERS SPECIFIC TO GRAPHIC DISPLAY SYSTEM
CC
CC          VIDEO: GRAPHIC SUBROUTINE PACKAGE "GSPNAME"
CC          NULL: GRAPHIC SUBROUTINE PACKAGE "NULL" VARIABLE
CC          I2250: GRAPHIC SUBROUTINE PACKAGE DEVICE NAME
CC          IGDS1: RESIDENT GRAPHIC DATA SET ADDRESS
CC          IGDS2: HISTORICAL OVERLAY GRAPHIC DATASET ADDRESS
CC          IGDS3: PREDICTIVE OVERLAY GRAPHIC DATA SET ADDRESS
CC          ATTN: PRIMARY ATTENTION LEVEL BASE ADDRESS
C           KEY (ALL): RELATIVE ADDRESSES OF SPECIFIEC GRAPHIC ORDERS
```

```
C                    IN IGDS2 HISTORICAL FILE
C          TO: SCREEN LOCATION OF 9:00 A.M.
C          DXDT: INCHES PER HOUR ON DISPLAY SCREEN
C          SLO: VERTICAL ORIGIN FOR LOWER LIMIT OF SALES ON SCREEN
C          VLO: VERTICAL ORIGIN FOR "0" FOR VOLUME ON SCREEN
C
C
C-----DEFINE LOCAL STORAGE
C
      LOGICAL*1 BYT1(4),BYT2(4)
      INTEGER*4 ENT
      INTEGER*4 IKEY
      INTEGER*4 ICRVL
      INTEGER*4 WORD1,WORD2
      REAL*4 YLOC
C
C-----DEFINE RELEVANT EQUIVALENT STORAGE LOCATIONS
C
      EQUIVALENCE (WORD1,BYT1(1)),(WORD2,BYT2(1))
C
C          (1) CREATE AN ATTENTION LEVEL FOR 2250, ATTN.
C
      CALL CRATL(I2250,ATTN)
C
C          (2) ENABLE ATTENTION SOURCES FOR "ATTN", AS FOLLOWS:
C             BUTTON 0 CAUSES ONE DATA INTERRPT TO BE PROCESSED
C             BUTTON 1 CAUSES TEN DATA INTERRUPTS TO BE PROCESSED
C             BUTTON 2 CAUSES 100 DATA INTERRUPTS TO BE PROCESSED
C             ATTENTION 34 IS THE LIGHT-PEN
C
      CALL ENATN(ATTN,0,1,2,34)
C
C          (3) LIGHT ACTIVE BUTTONS
C
      CALL MLITS(I2250,3)
      CALL SLPAT(IGDS1,1)
      CALL SLPAT(IGDS2,2)
      CALL SLPAT(IGDS3,2)
C
C          (4) GENERATE A TABLE IN IGDS1 TO BE DISPLAYED ON SCREEN.
C
      CALL STPOS(IGDS1,8.75,0.5)
      CALL PLINE(IGDS1,8.75,6.25)
      CALL PLINE(IGDS1,11.5,6.25)
      CALL STPOS(IGDS1,11.5,6.0)
      CALL PLINE(IGDS1,8.75,6.0)
      CALL PTEXT(IGDS1,'ACTIVE OPTIONS',14,NULL,NULL,NULL,8.90,6.10)
      CALL PTEXT(IGDS1,'FORECAST',8,44,  NULL,NULL,8.90,5.716)
```

```
      CALL PTEXT(IGDS1,'*',1,45,NULL,2,11.0,5.716)
      CALL PTEXT(IGDS1,'INTFACE',7,NULL,NULL,NULL,8.90,5.482)
      CALL PTEXT(IGDS1,'0-2',3,NULL,NULL,NULL,11.0,5.482)
      CALL PTEXT(IGDS1,'SHUTDOWN',8,46,  NULL,NULL,8.90,5.248)
      YLOC = 5.248
      DO 300 ENT = 1,19,1
      IF(MSTB1(ENT))300,300,210
  210 YLOC = YLOC- 0.234
      WORD1 = MSTB2(ENT)
      BYT2(1) = BYT1(1)
      WORD1 = MSTB3(ENT)
      BYT2(2) = BYT1(1)
      WORD1 = MSTB4(ENT)
      BYT2(3) = BYT1(1)
      CALL PTEXT(IGDS1,WORD2,3,ENT, NULL,NULL,8.90,YLOC)
      ICRVL = 50 + ENT
      CALL PTEXT(IGDS1,'*',1,ICRVL,IKEY,2,11.0,YLOC)
  300 CONTINUE
      CALL EXEC(IGDS1)
      CALL INCL(IGDS1)
      CALL OMIT(IGDS1,39)
      RETURN
C
CC        THIS IS A TRANSIENT ROUTINE WHICH MAY BE OVERLAID AFTER
C         EXECUTION HAS COMPLETED.
C
      END
C
C


      SUBROUTINE INIT6
C
CC        SUBROUTINE INIT6:
C
CC             1. INITIALLIZES ALL HISTORICAL GRAPHIC OVERLAY FILES
C         AND RELEVANT PARAMETERS;
C
CC             2. INITIALLIZES ALL PREDICTIVE GRAPHIC OVERLAY FILES
C         AND RELEVANT PARAMETERS ON THOSE FILES.
C
C-----DEFINE GLOBAL STORAGE
C
      INTEGER*4 MSTB1,MSTB2,MSTB3,MSTB4
      COMMON /MSTB/ MSTB1(19),MSTB2(19),MSTB3(19),MSTB4(19)
C
CC       MSTB: MASTER TABLE, CONTAINING CURRENT MONITORED COMMODITIES
C
```

```
C          MSTB1: NUMBER OF SIGNIFICANT CHARACTERS IN MNEMONIC
C          MSTB2: THIRD CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED, IF ANY
C          MSTB3: SECOND CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED
C          MSTB4: FIRST CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED
C
      INTEGER*4 VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      INTEGER*4 KEY,ATTN
      REAL*4 TO,DXDT,SLO,VLO
      COMMON /GRAPH/ VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      COMMON /GRAPH/ ATTN,KEY(5),TO,DXDT,SLO,VLO
C
C        GRAPH: CONTAINS PARAMETERS SPECIFIC TO GRAPHIC DISPLAY SYSTEM
C
C          VIDEO: GRAPHIC SUBROUTINE PACKAGE "GSPNAME"
C          NULL: GRAPHIC SUBROUTINE PACKAGE "NULL" VARIABLE
C          I2250: GRAPHIC SUBROUTINE PACKAGE DEVICE NAME
C          IGDS1: RESIDENT GRAPHIC DATA SET ADDRESS
C          IGDS2: HISTORICAL OVERLAY GRAPHIC DATASET ADDRESS
C          IGDS3: PREDICTIVE OVERLAY GRAPHIC DATA SET ADDRESS
C          ATTN: PRIMARY ATTENTION LEVEL BASE ADDRESS
C          KEY (ALL): RELATIVE ADDRESSES OF SPECIFIEC GRAPHIC ORDERS
C                     IN IGDS2 HISTORICAL FILE
C          TO: SCREEN LOCATION OF 9:00 A.M.
C          DXDT: INCHES PER HOUR ON DISPLAY SCREEN
C          SLO: VERTICAL ORIGIN FOR LOWER LIMIT OF SALES ON SCREEN
C          VLO: VERTICAL ORIGIN FOR "0" FOR VOLUME ON SCREEN
C
      INTEGER*4 IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
      COMMON /SYSIO/IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
C
C        SYSIO: SYSTEM INPUT/OUTPUT UNIT DESIGNATIONS
C
C          IO1: SYSTEM PRINTER
C          IO2: SYSTEM CARD READER
C          IO3: EXCHANGE INTERFACE INPUT FILE
C          IO4: GRAPHIC DISPLAY OUTPUT UNIT
C          IO5: RAW INFORMATION RECORDING UNIT (SEQUENTIAL)
C          IO6: HISTORICAL GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C          IO7: PREDICTIVE GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C          IO8: SYSTEM "SAVE" FILE WHEN SHUT DOWN (SEQUENTIAL)
C
      INTEGER*4 BLOCK,SUBBLK
      COMMON/BLOCK/BLOCK(522),SUBBLK(5)
C
C        BLOCK: PRIMARY SYSTEM INPUT/OUTPUT BUFFERS
C
C          BLOCK: SHARED HISTORICAL AND PREDICTIVE DIRECT-ACCESS
C          SUBBLK: RAW HISTORICAL RECORDER OUTPUT BUFFER
```

```
C
      INTEGER*4 INC40,INC50,MXBLK6,MXBLK7
      COMMON /DISKIO/ INC40,INC50,MXBLK6,MXBLK7
C
CCCC
CCCC     DISKIO: SPECIFIES DIRECT-ACCESS I/O POINTERS AND GRAPHIC
CCCC             DISPLAY BUFFER SIZES:
CCCC
CCCC       INC40: POINTS TO NEXT RECORD TO BE READ ON I06
CCCC       INC50: POINTS TO NEXT RECORD TO BE READ ON I07
CCCC       MXBLK6: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I06 FILE
CCCC       MXBLK7: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I07 FILE
CC
      INTEGER*4 STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
      COMMON /SYSSTE/ STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
C
CCCC     SYSSTE: SYSTEM STATE INDICATORS
CCCC
CCCC       STAT1:INDICATES NUMBER OF TIMES LEFT TO RECEIVE DATA (TEST)
CCCC       STAT2: INDICATES PREDICTIVE STATE ACTIVE IF EQUAL TO 2
CCCC       STAT3: INDICATES TYPE OF TRANSACTION ON EXCHANGE INTERFACE
CCCC          =2: ASK;
CCCC          =3: BID
CCCC          =4: SALE
CCCC          =5: VOLUME REPORT
CCCC       STAT5: INDICATES MASTER TABLE INDEX OF ACTIVE HISTORICAL OVERLAY
CCCC       STAT6: INDICATES WHETHER EXCHANGE INTERFACE OPEN OR CLOSED
CCCC       STAT7: MASTER TABLE INDEX OF ACTIVE PREDICTIVE DISPLAY
CCCC
CC-----DEFINE LOCAL STORAGE
C
      LOGICAL*1 DSPLY(5)
      INTEGER*4 BLOCKS(74)
      INTEGER*4 HOLD(512)
      INTEGER*4 HOLDS(64)
      INTEGER*4 GDALN
      INTEGER*4 INTIO
      INTEGER*4 KEYT
      INTEGER*4 TITLE(7),ENT,IKEY,II
      INTEGER*4 TMBS1
      REAL*4 LW1,LW2,HGH1,HGH2,YLOC,LABEL
      REAL*4 RTMP1(4),RTMP2(4)
      REAL*4 TMBS,LSTM
      REAL*4 ALPHA
      REAL*4 DVDY,DSDY
      REAL*4 ALPHA1
C
```

```
C-----DEFINE RELEVANT EQUIVALENT STORAGE LOCATIONS
C
      EQUIVALENCE (TMBS1,BLOCK(1))
      EQUIVALENCE (ALPHA1,BLOCK(2))
      EQUIVALENCE (DVDY,DSDY)
      EQUIVALENCE (BLOCK(1),BLOCKS(1))
      EQUIVALENCE (TMBS,BLOCK(5))
      EQUIVALENCE (LSTM,BLOCK(4))
      EQUIVALENCE (RTMP1(1),BLOCK(7))
      EQUIVALENCE (BLOCK(7),ALPHA)
      EQUIVALENCE (GDALN,BLOCK(6))
      EQUIVALENCE (LW1,BLOCK(7))
      EQUIVALENCE (BLOCK(8),HGH1,DYDS)
      EQUIVALENCE (BLOCK(9),LW2)
      EQUIVALENCE (BLOCK(10),HGH2,DYDV)
      EQUIVALENCE (BLOCK(11),HOLD(1),HOLDS(1))
C
C-----DEFINE ACCESSED DISK FILES
C
      DEFINE FILE 40(19,522,U,INC40)
      DEFINE FILE 50(19,74,U,INC50)
C
C-----LOOP THROUGH ENTIRE MASTER TABLE, SKIPPING MSTB1 = 0 ENTRIES.
C
      DO 200 ENT = 1,19,1
      IF(MSTB1(ENT))200,200,9
C
C----------->  BEGIN INITIALLIZATION OF IO7 PREDICTIVE FILE <----------
C
C          (1) GET INFORMATION FROM IO7, RESET AND WRITE BACK TO DISK
C
    9 READ(IO7'ENT)BLOCKS
      DO 10 II = 1,7
      TITLE(II) = BLOCK(II + 10)
   10 BLOCK(II+10) = 0
      DO 11 II = 1,4,1
   11 RTMP2(II) = RTMP1(II)
      DO 12 II = 8,10,1
      BLOCK(II) = BLOCK(II + 10)
   12 BLOCK(II+10) = 0
      ALPHA = ALPHA1
      ALPHA1 = 0.0
      LSTM = TO
      TMBS = 3600.0 / (FLOAT(TMBS1) * DXDT)
      TMBS1 = 0
      CALL RESET(IGDS3)
      CALL STPOS(IGDS3,0.0,0.0)
      CALL EXEC(IGDS3)
```

```
      CALL GSPRD(IGDS3,HOLDS,MXBLK7,2,GDALN)
      IF(GDALN)15,15,14
   15 GDALN = MXBLK7
   14 WRITE(IO7'ENT)BLOCKS
C
C----------->  END   INITIALLIZATION OF IO7 PREDICTIVE FILE <----------
C
C----------->  BEGIN INITIALLIZATION OF IO6 HISTORICAL FILE <----------
C
C          (3) RESET DATA SET AND GENERATE TITLE
C
      CALL RESET(IGDS2)
      CALL IDPOS(IGDS2,0.0,0.0)
      CALL PTEXT(IGDS2,TITLE,28,NULL,NULL,NULL,3.950,10.810)
C
C          (4) GENERATE DUMMY "LAST ACTION ENTRIES AND ASSIGN CORRELATION
C
C                    CORR 1:  TRANSACTION TYPE
C                    CORR 2:  PRICE
C                    CORR 3:  HOURS
C                    CORR 4:  MINUTES
C                    CORR 5:  SECONDS
C
      CALL PTEXT(IGDS2,'    ',4,NULL,KEYT,NULL,9.925,9.800)
      KEY(1) = KEYT
      CALL PTEXT(IGDS2,'  .  ',5,NULL,KEYT,NULL,9.835,9.110)
      KEY(2) = KEYT
      CALL PTEXT(IGDS2,'  ',2,NULL,KEYT,NULL,9.310,8.500)
      KEY(3) = KEYT
      CALL PTEXT(IGDS2,'  ',2,NULL,KEYT,NULL,9.720,8.500)
      KEY(4) = KEYT
      CALL PTEXT(IGDS2,'  ',2,NULL,KEYT,NULL,10.048,8.500)
      KEY(5) = KEYT
C
C          (5) READ I/O BLOCK INTO CORE FROM IO6
C
      READ(IO6'ENT)BLOCK
      DO 210 II = 1,4
  210 RTMP1(II) = RTMP2(II)
C
C          (6) GENERATE Y-AXIS TIC MARK FOR SALES
C
      DSDY  = (HGH1 - LW1) / 4.0
      DO 300 II = 1,5,1
      LABEL = LW1 + DSDY  * FLOAT(II - 1)
      YLOC = 6.0 + FLOAT(II - 1)
      CALL BCNV(LABEL,DSPLY,102,5,2)
  300 CALL PTEXT(IGDS2,DSPLY,5,NULL,NULL,NULL,0.610,YLOC)
```

```
      DYDS = 1.0 / DSDY
C
C          (7) GENERATE Y-AXIS TIC MARK FOR VOLUME
C
      DVDY  = (HGH2 - LW2) / 4.0
      DO 400 II = 1,5,1
      LABEL = LW2 + DVDY  * FLOAT(II - 1)
      YLOC = 1.5 + FLOAT(II - 1)
      INTIO = IFIX(LABEL)
      CALL BCNV(INTIO,DSPLY,103,5)
  400 CALL PTEXT(IGDS2,DSPLY,5,NULL,IKEY,NULL,0.610,YLOC)
      DYDV = 1.0 / DVDY
      CALL EXEC(IGDS2)
      CALL INCL(IGDS2)
      CALL GSPRD(IGDS2,HOLD,MXBLK6,2,GDALN)
      IF(GDALN)410,410,420
  410 GDALN = MXBLK6
  420 LSTM = TO
C
C          (8) WRITE BLOCK BACK ONTO DISK IO6
C
      WRITE(IO6'ENT)BLOCK
C
C----------->  END   INITIALLIZATION OF IO6 HISTORICAL FILE <----------
C
      STAT5 = ENT
      STAT7 = ENT
C
C          (14) RECYCLE TO NEXT MSTB ENTRY, IF ANY
C
  200 CONTINUE
      CALL OMIT(IGDS2)
      CALL OMIT(IGDS3)
C
      RETURN
C
C          THIS IS A TRANSIENT ROUTINE WHICH MAY BE OVERLAID AFTER
C          EXECUTION HAS COMPLETED.
C
      END
C
C
```

```
      SUBROUTINE HIST
C
CC         SUBROUTINE HIST TRANSFERS A REQUESTED HISTORICAL GRAPHIC
CC         OVERLAY TO THE BUFFER OF THE GRAPHIC DISPLAY UNIT AND
C          CAUSES IT TO BE DISPLAYED OVER THE RESIDENT GRAPHIC DISPLAY.
C
C-----DEFINE GLOBAL STORAGE
C
      INTEGER*4 STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
      COMMON /SYSSTE/ STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
C
CC       SYSSTE: SYSTEM STATE INDICATORS
CC
CC         STAT1:INDICATES NUMBER OF TIMES LEFT TO RECEIVE DATA (TEST)
CC         STAT2: INDICATES PREDICTIVE STATE ACTIVE IF EQUAL TO 2
CC         STAT3: INDICATES TYPE OF TRANSACTION ON EXCHANGE INTERFACE
CC            =2: ASK;
CC            =3: BID
CC            =4: SALE
CC            =5: VOLUME REPORT
CC         STAT5: INDICATES MASTER TABLE INDEX OF ACTIVE HISTORICAL OVERLAY
CC         STAT6: INDICATES WHETHER EXCHANGE INTERFACE OPEN OR CLOSED
CC         STAT7: MASTER TABLE INDEX OF ACTIVE PREDICTIVE DISPLAY
C
      INTEGER*4 IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
      COMMON /SYSIO/IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
C
CC       SYSIO: SYSTEM INPUT/OUTPUT UNIT DESIGNATIONS
CC
CC         IO1: SYSTEM PRINTER
CC         IO2: SYSTEM CARD READER
CC         IO3: EXCHANGE INTERFACE INPUT FILE
CC         IO4: GRAPHIC DISPLAY OUTPUT UNIT
CC         IO5: RAW INFORMATION RECORDING UNIT (SEQUENTIAL)
CC         IO6: HISTORICAL GRAPHIC OVERLAY FILE (DIRECT ACCESS)
CC         IO7: PREDICTIVE GRAPHIC OVERLAY FILE (DIRECT ACCESS)
CC         IO8: SYSTEM "SAVE" FILE WHEN SHUT DOWN (SEQUENTIAL)
C
      INTEGER*4 VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      INTEGER*4 KEY,ATTN
      REAL*4 TO,DXDT,SLO,VLO
      COMMON /GRAPH/ VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      COMMON /GRAPH/ ATTN,KEY(5),TO,DXDT,SLO,VLO
C
CC       GRAPH: CONTAINS PARAMETERS SPECIFIC TO GRAPHIC DISPLAY SYSTEM
CC
CC         VIDEO: GRAPHIC SUBROUTINE PACKAGE "GSPNAME"
C          NULL: GRAPHIC SUBROUTINE PACKAGE "NULL" VARIABLE
```

```
C           I2250: GRAPHIC SUBROUTINE PACKAGE DEVICE NAME
C           IGDS1: RESIDENT GRAPHIC DATA SET ADDRESS
C           IGDS2: HISTORICAL OVERLAY GRAPHIC DATASET ADDRESS
C           IGDS3: PREDICTIVE OVERLAY GRAPHIC DATA SET ADDRESS
C           ATTN: PRIMARY ATTENTION LEVEL BASE ADDRESS
C           KEY (ALL): RELATIVE ADDRESSES OF SPECIFIEC GRAPHIC ORDERS
C                      IN IGDS2 HISTORICAL FILE
C           TO: SCREEN LOCATION OF 9:00 A.M.
C           DXDT: INCHES PER HOUR ON DISPLAY SCREEN
C           SLO: VERTICAL ORIGIN FOR LOWER LIMIT OF SALES ON SCREEN
C           VLO: VERTICAL ORIGIN FOR "0" FOR VOLUME ON SCREEN
C
      INTEGER*4 BLOCK,SUBBLK
      COMMON/BLOCK/BLOCK(522),SUBBLK(5)
C
C        BLOCK: PRIMARY SYSTEM INPUT/OUTPUT BUFFERS
C
C          BLOCK: SHARED HISTORICAL AND PREDICTIVE DIRECT-ACCESS
C          SUBBLK: RAW HISTORICAL RECORDER OUTPUT BUFFER
C
      INTEGER*4 INC40,INC50
       COMMON /DISKIO/ INC40,INC50,DISKIO(2)
C
C        DISKIO: SPECIFIES DIRECT-ACCESS I/O POINTERS AND GRAPHIC
C                DISPLAY BUFFER SIZES:
C
C          INC40: POINTS TO NEXT RECORD TO BE READ ON I06
C          INC50: POINTS TO NEXT RECORD TO BE READ ON I07
C          MXBLK6: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I06 FILE
C          MXBLK7: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I07 FILE
C
C
C-----DEFINE LOCAL STORAGE
C
      INTEGER*4 HOLD(512)
      INTEGER*4 GDALN
C
C-----DEFINE RELEVANT EQUIVALENT STORAGE LOCATIONS
C
      EQUIVALENCE (BLOCK(11),HOLD(1))
      EQUIVALENCE (BLOCK(6),GDALN)
C
C-----DEFINE ACCESSED FILES
C
      DEFINE FILE 40(19,522,U,INC40)
C
C         (1) READ BLOCK OF GRAPHIC ORDERS INTO CORE FROM I06
C
```

```
      READ(IO6'STAT5)BLOCK
C
CC         (2) SHUT OFF IMAGE ON SCREEN AND RESET G.D.O.A. COMPLETELY.
C
      CALL RESET(IGDS2)
      CALL IDPOS(IGDS2,0.0,0.0)
C
CC         (3) MOVE G.D.S. FROM "HOLD" TO IGDS3 G.D.O.A-
C
      CALL ORGEN(IGDS2,HOLD,GDALN)
C
CC         (4) MOVE G.D.S. TO 2250 BUFFER FROM G.D.O.A., AND INCLUDE
C
      CALL EXEC(IGDS2)
      CALL INCL(IGDS2)
C
CC         (5) RETURN TO MAINLINE
C
      RETURN
C
CCCC       THIS IS A TRANSIENT ROUTINE WHICH MAY BE OVERLAID AFTER
CCCC       EXECUTION HAS COMPLETED.
C
      END
CC


      SUBROUTINE ALPHA
C
CCC        SUBROUTINE ALPHA PERMITS THE USER TO CHANGE THE VALUE OF
CCC        THE SMOOTHING CONSTANT, ONE OF THOSE PARAMETERS GOVERNING
CCC        THE STABILITY AND RESPONSE OF THE EXPONENTIAL SMOOTHING
CCC        EQUATIONS USED BY THE SYSTEM.  THE SUBROUTINE CAUSES
CCC        A REQUESTED GRAPHIC ORDER FILE TO BE LOADED INTO CORE,
CCC        AND CAUSES THE INTERRUPT STRUCTURE TO CHANGE SUCH THAT
CCC        A NEW VALUE OF ALPHA MAY BE ENTERED FROM THE DISPLAY
CCC        DEVICE'S KEYBOARD.  THIS VALUE IS CONVERTED TO INTERNAL
CCC        REPRESENTATION AND WRITTEN BACK ONTO THE PREDICTIVE FILE
CCC        DIRECT ACCESS UNIT.
CC
C-----DEFINE GLOBAL STORAGE
C
      INTEGER*4 STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
      COMMON /SYSSTE/ STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
C
CC        SYSSTE: SYSTEM STATE INDICATORS
C
```

```
C          STAT1:INDICATES NUMBER OF TIMES LEFT TO RECEIVE DATA (TEST)
C          STAT2: INDICATES PREDICTIVE STATE ACTIVE IF EQUAL TO 2
C          STAT3: INDICATES TYPE OF TRANSACTION ON EXCHANGE INTERFACE
C              =2: ASK;
C              =3: BID
C              =4: SALE
C              =5: VOLUME REPORT
C          STAT5: INDICATES MASTER TABLE INDEX OF ACTIVE HISTORICAL OVERLAY
C          STAT6: INDICATES WHETHER EXCHANGE INTERFACE OPEN OR CLOSED
C          STAT7: MASTER TABLE INDEX OF ACTIVE PREDICTIVE DISPLAY
C
      INTEGER*4 IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
      COMMON /SYSIO/ IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
C
C        SYSIO: SYSTEM INPUT/OUTPUT UNIT DESIGNATIONS
C
C          IO1: SYSTEM PRINTER
C          IO2: SYSTEM CARD READER
C          IO3: EXCHANGE INTERFACE INPUT FILE
C          IO4: GRAPHIC DISPLAY OUTPUT UNIT
C          IO5: RAW INFORMATION RECORDING UNIT (SEQUENTIAL)
C          IO6: HISTORICAL GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C          IO7: PREDICTIVE GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C          IO8: SYSTEM "SAVE" FILE WHEN SHUT DOWN (SEQUENTIAL)
C
      INTEGER*4 VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3,ATTN,KEY
      REAL*4 TO,DXDT,SLO,VLO
      COMMON /GRAPH/ VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3,ATTN,KEY(5),TO
      COMMON /GRAPH/DXDT,SLO,VLO
C
C        GRAPH: CONTAINS PARAMETERS SPECIFIC TO GRAPHIC DISPLAY SYSTEM
C
C          VIDEO: GRAPHIC SUBROUTINE PACKAGE "GSPNAME"
C          NULL: GRAPHIC SUBROUTINE PACKAGE "NULL" VARIABLE
C          I2250: GRAPHIC SUBROUTINE PACKAGE DEVICE NAME
C          IGDS1: RESIDENT GRAPHIC DATA SET ADDRESS
C          IGDS2: HISTORICAL OVERLAY GRAPHIC DATASET ADDRESS
C          IGDS3: PREDICTIVE OVERLAY GRAPHIC DATA SET ADDRESS
C          ATTN: PRIMARY ATTENTION LEVEL BASE ADDRESS
C          KEY (ALL): RELATIVE ADDRESSES OF SPECIFIEC GRAPHIC ORDERS
C                     IN IGDS2 HISTORICAL FILE
C          TO: SCREEN LOCATION OF 9:00 A.M.
C          DXDT: INCHES PER HOUR ON DISPLAY SCREEN
C          SLO: VERTICAL ORIGIN FOR LOWER LIMIT OF SALES ON SCREEN
C          VLO: VERTICAL ORIGIN FOR "0" FOR VOLUME ON SCREEN
C
      INTEGER*4 INC40,INC50
      COMMON /DISKIO/ INC40,INC50,DISKIO(2)
```

```
C
C        DISKIO: SPECIFIES DIRECT-ACCESS I/O POINTERS AND GRAPHIC
C                DISPLAY BUFFER SIZES:
C
C          INC40: POINTS TO NEXT RECORD TO BE READ ON I06
C          INC50: POINTS TO NEXT RECORD TO BE READ ON I07
C          MXBLK6: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I06 FILE
C          MXBLK7: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I07 FILE
C
      INTEGER*4 BLOCK,SUBBLK
      COMMON/BLOCK/BLOCK(522),SUBBLK(5)
C
C        BLOCK: PRIMARY SYSTEM INPUT/OUTPUT BUFFERS
C
C          BLOCK: SHARED HISTORICAL AND PREDICTIVE DIRECT-ACCESS
C          SUBBLK: RAW HISTORICAL RECORDER OUTPUT BUFFER
C
C
C-----DEFINE LOCAL STORAGE
C
      LOGICAL*1 TEXT(4)
      INTEGER*4 BLOCKS(74)
      REAL*4 ALPHA
C
C-----DEFINE RELEVANT EQUIVALENT STORAGE LOCATIONS
C
      EQUIVALENCE (BLOCK(1),BLOCKS(1))
      EQUIVALENCE (BLOCK(7),ALPHA)
C
C-----DEFINE ACCESSED FILES
C
      DEFINE FILE 50(19,74,U,INC50)
C
C          (1) READ PERTINENT FILE FROM I07
C
      READ(I07'STAT7)BLOCKS
C
C          (2) MODIFY ATTENTION LEVELS,SOURCES, AND INSERT CURSOR,ETC.
C
      CALL DSATN(ATTN,0,1,2,34)
      CALL MLITS(I2250,2)
      CALL ENATL(ATTN)
      CALL CRATL(I2250,IATN,1)
      CALL ENATN(IATN,32)
      CALL ICURS(IGDS1,41)
      CALL RQATN(IATN,IRTN,2,NULL,32)
      CALL GSPRD(IGDS1,TEXT,4,1,NULL,41)
      CALL RCURS(IGDS1)
```

```
      CALL BCNV(TEXT,ALPHA,202,4,3)
      CALL DSATN(IATN,32)
      CALL ENATL(IATN)
      CALL CRATL(I2250,ATTN,1)
      CALL ENATN(ATTN,0,1,2,34)
      CALL MLITS(I2250,3)
C
C         (3) WRITE BLOCK BACK ON DISK AND RETURN
C
      WRITE(I07'STAT7)BLOCKS
      RETURN
C
C         THIS IS A TRANSIENT ROUTINE WHICH MAY BE OVERLAID AFTER
C         EXECUTION HAS COMPLETED.
C
      END
C
C


      SUBROUTINE TBASE
C
C         SUBROUTINE TBASE PERMITS THE USER TO CHANGE A SPECIFIC VALUE
C         OF A "TIME-BASE", A PARAMETER USED USED BY THE EXPONENTIAL
C         SMOOTHING EQUATIONS BY WHICH STATISTICAL FORECASTS ARE
C         EFFECTED.  THIS SUBROUTINE CAUSES A REQUESTED GRAPHIC ORDER
C         FILE TO BE LOADED INTO CORE, AND MODIFIES THE GRAPHIC SYSTEM
C         SUCH THAT ALPHAMERIC INFORMATION MAY BE ENTERED FROM THE
C         DISPLAY DEVICE KEYBOARD.  THIS INFORMATION IS THEN CONVERTED
C         TO AN INTERNAL REPRESENTATION OF THE TIMEBASE, AND RE-WRITTEN
C         BACK ONTO THE PREDICTIVE DIRECT ACCESS UNIT.
C
C-----DEFINE GLOBAL STORAGE
C
      INTEGER*4 STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
      COMMON /SYSSTE/ STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
C
C        SYSSTE: SYSTEM STATE INDICATORS
C
C          STAT1:INDICATES NUMBER OF TIMES LEFT TO RECEIVE DATA (TEST)
C          STAT2: INDICATES PREDICTIVE STATE ACTIVE IF EQUAL TO 2
C          STAT3: INDICATES TYPE OF TRANSACTION ON EXCHANGE INTERFACE
C              =2: ASK;
C              =3: BID
C              =4: SALE
C              =5: VOLUME REPORT
C          STAT5: INDICATES MASTER TABLE INDEX OF ACTIVE HISTORICAL OVERLAY
C          STAT6: INDICATES WHETHER EXCHANGE INTERFACE OPEN OR CLOSED
```

```
C
C          STAT7: MASTER TABLE INDEX OF ACTIVE PREDICTIVE DISPLAY
C
      INTEGER*4 IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
      COMMON /SYSIO/ IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
C
CC       SYSIO: SYSTEM INPUT/OUTPUT UNIT DESIGNATIONS
CC
CC         IO1: SYSTEM PRINTER
CC         IO2: SYSTEM CARD READER
CC         IO3: EXCHANGE INTERFACE INPUT FILE
CC         IO4: GRAPHIC DISPLAY OUTPUT UNIT
CC         IO5: RAW INFORMATION RECORDING UNIT (SEQUENTIAL)
CC         IO6: HISTORICAL GRAPHIC OVERLAY FILE (DIRECT ACCESS)
CC         IO7: PREDICTIVE GRAPHIC OVERLAY FILE (DIRECT ACCESS)
CC         IO8: SYSTEM "SAVE" FILE WHEN SHUT DOWN (SEQUENTIAL)
C
      INTEGER*4 VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3,ATTN,KEY
      REAL*4 TO,DXDT,SLO,VLO
      COMMON /GRAPH/ VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3,ATTN,KEY(5),TO
      COMMON /GRAPH/DXDT,SLO,VLO
C
CC       GRAPH: CONTAINS PARAMETERS SPECIFIC TO GRAPHIC DISPLAY SYSTEM
CC
CC         VIDEO: GRAPHIC SUBROUTINE PACKAGE "GSPNAME"
CC         NULL: GRAPHIC SUBROUTINE PACKAGE "NULL" VARIABLE
CC         I2250: GRAPHIC SUBROUTINE PACKAGE DEVICE NAME
CC         IGDS1: RESIDENT GRAPHIC DATA SET ADDRESS
CC         IGDS2: HISTORICAL OVERLAY GRAPHIC DATASET ADDRESS
CC         IGDS3: PREDICTIVE OVERLAY GRAPHIC DATA SET ADDRESS
CC         ATTN: PRIMARY ATTENTION LEVEL BASE ADDRESS
CC         KEY (ALL): RELATIVE ADDRESSES OF SPECIFIEC GRAPHIC ORDERS
CC                    IN IGDS2 HISTORICAL FILE
CC         TO: SCREEN LOCATION OF 9:00 A.M.
CC         DXDT: INCHES PER HOUR ON DISPLAY SCREEN
CC         SLO: VERTICAL ORIGIN FOR LOWER LIMIT OF SALES ON SCREEN
CC         VLO: VERTICAL ORIGIN FOR "0" FOR VOLUME ON SCREEN
C
      INTEGER*4 INC40,INC50
      COMMON /DISKIO/ INC40,INC50,DISKIO(2)
C
CC       DISKIO: SPECIFIES DIRECT-ACCESS I/O POINTERS AND GRAPHIC
CC               DISPLAY BUFFER SIZES:
CC
CC         INC40: POINTS TO NEXT RECORD TO BE READ ON IO6
CC         INC50: POINTS TO NEXT RECORD TO BE READ ON IO7
CC         MXBLK6: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO IO6 FILE
CC         MXBLK7: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO IO7 FILE
C
```

```
      INTEGER*4 BLOCK,SUBBLK
      COMMON/BLOCK/BLOCK(522),SUBBLK(5)
C
CC       BLOCK: PRIMARY SYSTEM INPUT/OUTPUT BUFFERS
CC
C          BLOCK: SHARED HISTORICAL AND PREDICTIVE DIRECT-ACCESS
C          SUBBLK: RAW HISTORICAL RECORDER OUTPUT BUFFER
C
C
C-----DEFINE LOCAL STORAGE
C
      LOGICAL*1 TEXT(4)
      INTEGER*4 BLOCKS(74)
      REAL*4 TBASE
C
C-----DEFINE RELEVANT EQUIVALENT STORAGE LOCATIONS
C
      EQUIVALENCE (BLOCK(1),BLOCKS(1))
      EQUIVALENCE (BLOCK(5),TBASE)
C
C-----DEFINE ACCESSED FILES
C
      DEFINE FILE 50(19,74,U,INC50)
C
CC        (1) READ PERTINENT FILE FROM I07
C
      READ(I07'STAT7)BLOCKS
C
CC        (2) MODIFY ATTENTION LEVELS,SOURCES, AND INSERT CURSOR,ETC.
C
      CALL DSATN(ATTN,0,1,2,34)
      CALL MLITS(I2250,2)
      CALL ENATL(ATTN)
      CALL CRATL(I2250,IATN,1)
      CALL ENATN(IATN,32)
      CALL ICURS(IGDS1,43)
      CALL RQATN(IATN,IRTN,2,NULL,32)
      CALL GSPRD(IGDS1,TEXT,3,1,NULL,43)
      CALL RCURS(IGDS1)
      CALL BCNV(TEXT,IHOLD,203,3)
C
      TBASE = 3.6E+03 / (FLOAT(IHOLD) * DXDT)
      CALL DSATN(IATN,32)
      CALL ENATL(IATN)
      CALL CRATL(I2250,ATTN,1)
      CALL ENATN(ATTN,0,1,2,34)
      CALL MLITS(I2250,3)
C
```

```
C          (3) WRITE BLOCK BACK ON DISK AND RETURN
C
      WRITE(IO7'STAT7)BLOCKS
      RETURN
C
C          THIS IS A TRANSIENT ROUTINE WHICH MAY BE OVERLAID AFTER
C          EXECUTION HAS COMPLETED.
      END
C
C


      SUBROUTINE PRED
C
C          SUBROUTINE PRED RETRIEVES A REQUESTED PREDICTIVE OVERLAY
C          FROM THE PREDICTIVE FILE DIRECT-ACCESS UNIT AND CAUSES IT TO
C          BE DISPLAYED OVER THE RESIDENT  GRAPHIC DISPLAY.
C
C-----DEFINE GLOBAL STORAGE
C
      INTEGER*4 STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
      COMMON /SYSSTE/ STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
C
C        SYSSTE: SYSTEM STATE INDICATORS
C
C          STAT1:INDICATES NUMBER OF TIMES LEFT TO RECEIVE DATA (TEST)
C          STAT2: INDICATES PREDICTIVE STATE ACTIVE IF EQUAL TO 2
C          STAT3: INDICATES TYPE OF TRANSACTION ON EXCHANGE INTERFACE
C              =2: ASK;
C              =3: BID
C              =4: SALE
C              =5: VOLUME REPORT
C          STAT5: INDICATES MASTER TABLE INDEX OF ACTIVE HISTORICAL OVERLAY
C          STAT6: INDICATES WHETHER EXCHANGE INTERFACE OPEN OR CLOSED
C          STAT7: MASTER TABLE INDEX OF ACTIVE PREDICTIVE DISPLAY
C
      INTEGER*4 IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
      COMMON /SYSIO/ IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
C
C        SYSIO: SYSTEM INPUT/OUTPUT UNIT DESIGNATIONS
C
C          IO1: SYSTEM PRINTER
C          IO2: SYSTEM CARD READER
C          IO3: EXCHANGE INTERFACE INPUT FILE
C          IO4: GRAPHIC DISPLAY OUTPUT UNIT
C          IO5: RAW INFORMATION RECORDING UNIT (SEQUENTIAL)
C          IO6: HISTORICAL GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C          IO7: PREDICTIVE GRAPHIC OVERLAY FILE (DIRECT ACCESS)
```

```
C          I08: SYSTEM "SAVE" FILE WHEN SHUT DOWN (SEQUENTIAL)
C
      INTEGER*4 VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3,ATTN,KEY
      REAL*4 TO,DXDT,SLO,VLO
      COMMON /GRAPH/ VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3,ATTN,KEY(5),TO
      COMMON /GRAPH/DXDT,SLO,VLO
C
CC       GRAPH: CONTAINS PARAMETERS SPECIFIC TO GRAPHIC DISPLAY SYSTEM
CC
CC         VIDEO: GRAPHIC SUBROUTINE PACKAGE "GSPNAME"
CC         NULL: GRAPHIC SUBROUTINE PACKAGE "NULL" VARIABLE
CC         I2250: GRAPHIC SUBROUTINE PACKAGE DEVICE NAME
CC         IGDS1: RESIDENT GRAPHIC DATA SET ADDRESS
CC         IGDS2: HISTORICAL OVERLAY GRAPHIC DATASET ADDRESS
CC         IGDS3: PREDICTIVE OVERLAY GRAPHIC DATA SET ADDRESS
CC         ATTN: PRIMARY ATTENTION LEVEL BASE ADDRESS
CC         KEY (ALL): RELATIVE ADDRESSES OF SPECIFIEC GRAPHIC ORDERS
CC                    IN IGDS2 HISTORICAL FILE
CC         TO: SCREEN LOCATION OF 9:00 A.M.
CC         DXDT: INCHES PER HOUR ON DISPLAY SCREEN
CC         SLO: VERTICAL ORIGIN FOR LOWER LIMIT OF SALES ON SCREEN
CC         VLO: VERTICAL ORIGIN FOR "0" FOR VOLUME ON SCREEN
C
      INTEGER*4 INC40,INC50
      COMMON /DISKIO/ INC40,INC50,DISKIO(2)
C
CC       DISKIO: SPECIFIES DIRECT-ACCESS I/O POINTERS AND GRAPHIC
CC               DISPLAY BUFFER SIZES:
CC
CC         INC40: POINTS TO NEXT RECORD TO BE READ ON I06
CC         INC50: POINTS TO NEXT RECORD TO BE READ ON I07
CC         MXBLK6: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I06 FILE
CC         MXBLK7: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I07 FILE
C
      INTEGER*4 BLOCK,SUBBLK
      COMMON/BLOCK/BLOCK(522),SUBBLK(5)
C
CC       BLOCK: PRIMARY SYSTEM INPUT/OUTPUT BUFFERS
CC
CC         BLOCK: SHARED HISTORICAL AND PREDICTIVE DIRECT-ACCESS
CC         SUBBLK: RAW HISTORICAL RECORDER OUTPUT BUFFER
CC
C-----DEFINE LOCAL STORAGE
C
      LOGICAL*1 DSPLY(4)
      INTEGER*4 HOLDS(64)
      INTEGER*4 BLOCKS(74)
```

```
      INTEGER*4 GDALN
      REAL*4 ALPHA,TBASE
C
C-----DEFINE RELEVANT EQUIVALENT STORAGE LOCATIONS
C
      EQUIVALENCE (BLOCK(1),BLOCKS(1))
      EQUIVALENCE (BLOCK(5),TBASE)
      EQUIVALENCE (BLOCK(6),GDALN)
      EQUIVALENCE (BLOCK(7),ALPHA)
      EQUIVALENCE (BLOCK(11),HOLDS(1))
C
C-----DEFINE ACCESSED FILES
C
      DEFINE FILE 50(19,138,U,INC50)
C
CC        (1) LOAD FILE INDICATED BY STAT7 INTO CORE
C
      READ(IO7'STAT7)BLOCKS
C
CC        (2) TRANSMIT ORDERS TO GDOA
C
      CALL RESET(IGDS3)
      CALL ORGEN(IGDS3,HOLDS,GDALN)
C
CC        (3) TRANSMIT GDOA TO BUFFER
C
      CALL EXEC(IGDS3)
C
CC        (4) DISPLAY OVERLAY
C
      CALL INCL(IGDS3)
C
CC        (5) UPDATE AND DISPLAY "ALPHA" AND "TBASE" BLOCKS
C
      CALL BCNV(ALPHA,DSPLY,102,4,3)
      CALL IDPOS(IGDS1,0.0,0.0)
      CALL PTEXT(IGDS1,DSPLY,4,41,NULL,3,9.850,7.850)
      IHOLD = IFIX(3.6E+03 / (TBASE * DXDT))
      CALL BCNV(IHOLD,DSPLY,103,3)
      CALL PTEXT(IGDS1,DSPLY,3,43,NULL,3,9.850,7.225)
      CALL INCL(IGDS1,39)
      CALL IDPOS(IGDS1,0.0,0.0)
C
      RETURN
C
CC        THIS IS A TRANSIENT ROUTINE WHICH MAY BE OVERLAID AFTER
CC        EXECUTION HAS COMPLETED.
C
```

```
      END
C
C
      SUBROUTINE DOWN
C
CC        SUBROUTINE DOWN SHUTS THE SYSTEM DOWN IN AN ORDERLY FASHION:
CC
CC          A. DISPLAY MESSAGE ON I2250, RESETTING IGDS1 COMPLETELY.
CC
CC          B. REWIND EXCHANGE-INTERFACE SIMULATOR FILE
CC
CC          C. TRANSFER MASTER TABLE TO SYSTEM PRINTER AND TO
CC             SEQUENTIAL FILE.
CC
CC          D. TRANSFER ALL SYSTEM DIRECT ACCESS FILES (NAMELY,
CC             HISTORICAL AND PREDICTIVE GRAPHIC OVERLAYS) TO
CC             SEQUENTIAL FILE.
CC
CC          E. TRANSFER SUITABLE RE-INITIALLIZATION INFORMATION TO
CC             SYSTEM PRINTER AND TO SYSTEM CARD PUNCH.
CC
CC          F. DE-ACTIVATE ATTENTION LEVELS AND SOURCES
CC
CC          G. DE-ACTIVATE 2250 DISPLAY UNIT
CC
CC          H. DE-ACTIVATE GRAPHIC SUBROUTINE PACKAGE.
CC
C-----DEFINE GLOBAL STORAGE
C
      INTEGER*4 MSTB1,MSTB2,MSTB3,MSTB4
      COMMON /MSTB/MSTB1(19),MSTB2(19),MSTB3(19),MSTB4(19)
C
CC       MSTB: MASTER TABLE, CONTAINING CURRENT MONITORED COMMODITIES
CC
CC         MSTB1: NUMBER OF SIGNIFICANT CHARACTERS IN MNEMONIC
CC         MSTB2: THIRD CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED, IF ANY
CC         MSTB3: SECOND CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED
CC         MSTB4: FIRST CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED
C
      INTEGER*4 IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
      COMMON /SYSIO/IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
C
CCC       SYSIO: SYSTEM INPUT/OUTPUT UNIT DESIGNATIONS
CCC
C           IO1: SYSTEM PRINTER
```

```
C           I02: SYSTEM CARD READER
C           I03: EXCHANGE INTERFACE INPUT FILE
C           I04: GRAPHIC DISPLAY OUTPUT UNIT
C           I05: RAW INFORMATION RECORDING UNIT (SEQUENTIAL)
C           I06: HISTORICAL GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C           I07: PREDICTIVE GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C           I08: SYSTEM "SAVE" FILE WHEN SHUT DOWN (SEQUENTIAL)
C
      INTEGER*4 VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      INTEGER*4 ATTN,KEY
      REAL*4 TO,DXDT,SLO,VLO
      COMMON /GRAPH/VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      COMMON /GRAPH/ ATTN,KEY(5),TO,DXDT,SLO,VLO
C
C        GRAPH: CONTAINS PARAMETERS SPECIFIC TO GRAPHIC DISPLAY SYSTEM
C
C          VIDEO: GRAPHIC SUBROUTINE PACKAGE "GSPNAME"
C          NULL: GRAPHIC SUBROUTINE PACKAGE "NULL" VARIABLE
C          I2250: GRAPHIC SUBROUTINE PACKAGE DEVICE NAME
C          IGDS1: RESIDENT GRAPHIC DATA SET ADDRESS
C          IGDS2: HISTORICAL OVERLAY GRAPHIC DATASET ADDRESS
C          IGDS3: PREDICTIVE OVERLAY GRAPHIC DATA SET ADDRESS
C          ATTN: PRIMARY ATTENTION LEVEL BASE ADDRESS
C          KEY (ALL): RELATIVE ADDRESSES OF SPECIFIEC GRAPHIC ORDERS
C                     IN IGDS2 HISTORICAL FILE
C          TO: SCREEN LOCATION OF 9:00 A.M.
C          DXDT: INCHES PER HOUR ON DISPLAY SCREEN
C          SLO: VERTICAL ORIGIN FOR LOWER LIMIT OF SALES ON SCREEN
C          VLO: VERTICAL ORIGIN FOR "0" FOR VOLUME ON SCREEN
C
      INTEGER*4 DATE
      COMMON /DATE/ DATE(8)
C
C         DATE: CONTAINS CURRENT DATE
C
      INTEGER*4 BLOCK,SUBBLK
      COMMON/BLOCK/BLOCK(522),SUBBLK(5)
C
C        BLOCK: PRIMARY SYSTEM INPUT/OUTPUT BUFFERS
C
C          BLOCK: SHARED HISTORICAL AND PREDICTIVE DIRECT-ACCESS
C          SUBBLK: RAW HISTORICAL RECORDER OUTPUT BUFFER
C
      INTEGER*4 INC40,INC50
      COMMON /DISKIO/ INC40,INC50,DISKIO(2)
C
C        DISKIO: SPECIFIES DIRECT-ACCESS I/O POINTERS AND GRAPHIC
C                DISPLAY BUFFER SIZES:
```

```
C
CCC          INC40: POINTS TO NEXT RECORD TO BE READ ON I06
CCC          INC50: POINTS TO NEXT RECORD TO BE READ ON I07
CCC          MXBLK6: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I06 FILE
CCC          MXBLK7: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I07 FILE
C
      INTEGER*4 HOURS,HSEC
      COMMON /INPUT/ HOURS,HSEC,INPUT(3)
C
CCC       INPUT: INPUT BUFFER FOR EXCHANGE-INTERFACE
CCC
CCC          HOURS: HOUR OF DAY OF TRANSMISSION, IN 24-HOUR TIME
CCC          HSEC: HUNDRETHS SECONDS INTO HOUR
CCC          INDEX: HASH-ADDRESS OF MNEMONIC IN MASTER TABLE
CCC          TRANS: REAL REPRESENTATION OF "NUMERICAL VALUE" OF INPUT
CCC          TIME: TIME OF TRANSMISSION EXPRESSED IS DISPLAY-ORIENTED INCHES
CCC
CC-----DEFINE LOCAL STORAGE
C
      LOGICAL*1 DSPLY(5)
      INTEGER*4 BLOCKS(74)
      INTEGER*4 ENT,II
C
C-----DEFINE RELEVANT EQUIVALENCED STORAGE LOCATIONS
C
      EQUIVALENCE (BLOCK(1),BLOCKS(1))
C
C-----DEFINE ACCESSED FILES
C
      DEFINE FILE 50(19,74,U,INC50)
      DEFINE FILE 40(19,522,U,INC40)
C
CCC
CCC          (1) DEFINE SYSTEM CARD PUNCH
C
      I09 = 7
C
CCC          (2) RESET IGDS1 AND DISPLAY MESSAGE IN LARGE TEXT
C
      CALL RESET(IGDS1)
      CALL IDPOS(IGDS1,0.0,0.0)
      CALL SCHAM(IGDS1,2)
      CALL PTEXT(IGDS1,'SYSTEM SHUTDOWN IN PROCESS.',27,NULL,NULL,NULL,
     12.0,8.0)
      CALL PTEXT(IGDS1,'ALL TABLES AND DISK FILES',25,NULL,NULL,NULL,
     12.0,7.5)
      CALL PTEXT(IGDS1,'ARE RECORDED ON LOGICAL DEVICE',30,NULL,NULL,
```

```
     1NULL,2.0,7.0)
      CALL BCNV(IO8,DSPLY,103,2)
      CALL PTEXT(IGDS1,DSPLY,2,NULL,NULL,NULL,4.0,6.5)
      CALL PTEXT(IGDS1,'SEE PRINTER SUMMARY',19,NULL,NULL,NULL,2.0,6.0)
      CALL PTEXT(IGDS1,'TONE WILL COMPLETE SHUTDOWN',27,NULL,NULL,NULL,
     12.0,5.5)
      CALL EXEC(IGDS1)
      CALL OMIT(IGDS2)
      CALL OMIT(IGDS3)
      CALL INCL(IGDS1)
C
CC         (3) REWIND EXCHANGE INTERFACE SIMULATOR FILE
C
      REWIND IO3
C
CC         (4) WRITE MASTER TABLE ON SYSTEM PRINTER, AND ON
CC         SEQUENTIAL FILE, IO8, FILE NUMBER 1
CC
CC             A. MSG TO PRINTER
C
      WRITE(IO1,1000)HOURS,HSEC, (DATE(II),II=1,8)
 1000 FORMAT('1',T26,'SYSTEM SHUTDOWN INITIATED.',/,T30,'TIME = ',I2,':'
     1,I7,T50,'DATE = ',8A1,//)
C
CC             B. MASTER TABLE TO PRINTER AND FILE "IO8"
C
      WRITE(IO1,2000)
 2000 FORMAT('1',T23,'MASTER TABLE AT SHUTDOWN',/)
    2 DO 100 II = 1,19,1
      IF(MSTB1(II).GT.0)WRITE(6,2100)II,MSTB2(II),MSTB3(II),MSTB4(II)
 2100 FORMAT(T20,'HASH ADDRESS, ',I2,' CONTAINED MNEMONIC "',3A1,'"')
  100 WRITE(IO8)MSTB1,MSTB2,MSTB3,MSTB4
      ENDFILE IO8
C
CC         (5) TRANSFER ALL DIRECT-ACCESS FILES TO IO8 (SEQUENTIAL)
CC
CC             (A) END FILE IO5
C
      END FILE IO5
      REWIND IO5
C
CC             (A) TRANSFER IO6 TO IO8 AS 19 RECORDS OF FILE 2
C
      DO 220 II = 1,19
      READ(IO6'II)BLOCK
  220 WRITE(IO8)BLOCK
      END FILE IO8
C
```

```
C              (C) TRANSFER IO7 TO IO8 AS 19 RECORDS OF FILE 3;
CC                 IF MSTB1 IS NOT ZERO, PUNCH INITIALLIZATION
CC                 DATA TO CARDS FOR NEXT SYSTEM INITIALLIZATION
C
      DO 230 II =1,19
      READ(IO7'II)BLOCKS
      IF(MSTB1(II).EQ.0)GOTO 230
      ITMBS = IFIX(3600.0 / (TMBS * DXDT))
      WRITE(IO1,5000)MSTB2(II),MSTB3(II),MSTB4(II), BLOCKS(8),BLOCKS(9),
     XBLOCKS(10),BLOCKS(7),ITMBS
 5000 FORMAT(' STATISTICS FOR "',3A1,'": ',3E10.2,' ALPHA: ',F5.3,' TMBS
     1: ',I4)
      WRITE(IO9,5001)BLOCKS(8),BLOCKS(9),BLOCKS(10),BLOCKS(7),ITMBS,
     X MSTB2(II),MSTB3(II),MSTB4(II)
 5001 FORMAT(3(E9.2,1X),F5.3,1X,I3,1X,A3)
  230 WRITE(IO8)BLOCKS
      END FILE IO8
C
CC            (7) WRITE LOGICAL FILE NUMBERS ON IO8 ON SYSTEM PRINTER
C
      WRITE(IO1,3000)(IO8,II=1,3)
 3000 FORMAT('1',T23,'SYSTEM SAVE FILES AS FOLLOWS ON "SYSSAV":',/,' MAS
     1TER TABLE ON FT',I2,'F001',/,' GRAPHIC SEQUENCE IGDS2 ON FT',I2,'F
     2002',/,' GRAPHIC SEQUENCE IGDS3 ON FT',I2,'F003',/)
C
CC            (8) DE-ACTIVATE ALL GRAPHIC ATTENTION SOURCES.
C
      CALL MLITS(I2250,2)
      CALL ENATN(ATTN,1,-35)
C
CC            (9) DE-ACTIVATE 2250-1 UNIT
C
      CALL TMDEV(I2250)
C
CC            (11) DE-ACTIVATE GRAPHIC SUBROUTINE PACKAGE
C
      CALL TMGSP(VIDEO)
C
CC            (11) REWIND IO8 SEQUENTIAL FILE (IGNORED IF DISK FILE)
C
      REWIND IO8
C
CC            (12) MSG TO IO1
C
      WRITE(IO1,4000)
 4000 FORMAT(T30,'SYSTEM SHUTDOWN COMPLETED')
C
C-----RETURN TO MAINLINE
```

```
C
      RETURN
C
CCC       THIS IS A TRANSIENT ROUTINE WHICH MAY BE OVERLAID AFTER
CCC       EXECUTION HAS COMPLETED.
C
      END
CC


      SUBROUTINE MNITR
C
CCC       SUBROUTINE MNITR ACCEPTS USER REQUESTS FROM THE LIGHT PEN,
CCC       IN STANDARD MODE OF OPERATION, AND FROM THE PROGRAMMED
CCC       FUNCTION BOX FOR PROGRAM TEST CONTROL.
CCC
C-----DEFINE GLOBAL STORAGE
C
      INTEGER*4 VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      INTEGER*4 KEY,ATTN
      REAL*4 TO,DXDT,SLO,VLO
      COMMON /GRAPH/ VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      COMMON /GRAPH/ ATTN,KEY(5),TO,DXDT,SLO,VLO
C
CCC      GRAPH: CONTAINS PARAMETERS SPECIFIC TO GRAPHIC DISPLAY SYSTEM
CCCC
CCCC       VIDEO: GRAPHIC SUBROUTINE PACKAGE "GSPNAME"
CCCC       NULL: GRAPHIC SUBROUTINE PACKAGE "NULL" VARIABLE
CCCC       I2250: GRAPHIC SUBROUTINE PACKAGE DEVICE NAME
CCCC       IGDS1: RESIDENT GRAPHIC DATA SET ADDRESS
CCCC       IGDS2: HISTORICAL OVERLAY GRAPHIC DATASET ADDRESS
CCCC       IGDS3: PREDICTIVE OVERLAY GRAPHIC DATA SET ADDRESS
CCCC       ATTN: PRIMARY ATTENTION LEVEL BASE ADDRESS
CCCC       KEY (ALL): RELATIVE ADDRESSES OF SPECIFIEC GRAPHIC ORDERS
CCCC                  IN IGDS2 HISTORICAL FILE
CCCC       TO: SCREEN LOCATION OF 9:00 A.M.
CCCC       DXDT:  INCHES PER HOUR ON DISPLAY SCREEN
CCCC       SLO: VERTICAL ORIGIN FOR LOWER LIMIT OF SALES ON SCREEN
CCCC       VLO: VERTICAL ORIGIN FOR "0" FOR VOLUME ON SCREEN
C
      INTEGER*4 BRNCH
      COMMCN /SYSBRN/BRNCH
C
CCC      SYSBRN: SYSTEM BRANCH INDICATOR
CCCC
CCC       THIS INDICATOR IS SET BY SUBROUTINES TO SHIFT SYSTEM CONTROL
C         AFTER EXECUTION.
```

```
C
      INTEGER*4 STAT1,STAT2,STAT3,       STAT5,STAT6,STAT7
      COMMON /SYSSTE/STAT1,STAT2,STAT3,       STAT5,STAT6,STAT7
C
CC         SYSSTE: SYSTEM STATE INDICATORS
CC
CC           STAT1:INDICATES NUMBER OF TIMES LEFT TO RECEIVE DATA (TEST)
CC           STAT2: INDICATES PREDICTIVE STATE ACTIVE IF EQUAL TO 2
CC           STAT3: INDICATES TYPE OF TRANSACTION ON EXCHANGE INTERFACE
CC              =2: ASK;
CC              =3: BID
CC              =4: SALE
CC              =5: VOLUME REPORT
CC           STAT5: INDICATES MASTER TABLE INDEX OF ACTIVE HISTORICAL OVERLAY
CC           STAT6: INDICATES WHETHER EXCHANGE INTERFACE OPEN OR CLOSED
CC           STAT7: MASTER TABLE INDEX OF ACTIVE PREDICTIVE DISPLAY
C
C-----DEFINE LOCAL STORAGE
C
      INTEGER*4 ICRVL
      INTEGER*4 RTRN
      INTEGER*4 PEN(10)
C
CC          (1) REQUEST ATTENTION INFORMATION
C
   10 CALL SALRM(I2250)
      CALL RQATN(ATTN,RTRN,2,PEN,0,1,2,34)
C
CC          (2) IF NO ACTIVE BUTTON CAUSED ATTENTION, RECYCLE.
C
      IF(RTRN)100,10,120
C
CCC         (3) IF BUTTON 0 PRESSED, CHECK STAT6. IF 2, RECYLE. ELSE
CCC             SET STAT1 TO 1 AND BRANCH TO WULINE TO READ 1 TRANSACTION.
C
  100 GOTO(110,10),STAT6
  110 STAT1 = 1
      GOTO 150
C
CC          (4) IF LIGHT PEN DETECTED (#34) GOTO 200
C
  120 IF(34 - RTRN)10,200,130
C
CC          (5) IF BUTTON 0, 1, OR 2 PRESSED, SET STAT1 TO 1, 1/, 100,
CC             RESPECTIVELY, AND BRANCH TO RECVR VIA MAINLINE
CC             TO BEGIN ACCEPTING  SIMULATED INFORMATION UNITS FROM
CC             PRE-PROCESSED FILE ACROSS EXCHANGE INTERFACE
CC
```

```
C             IF STAT6 IS EQUAL TO 2, THEN EXCHANGE IS CLOSED, AND
C             RECYCLE TO REQUEST ATTENTION INFORMATION.  THIS IS
C             NECESSARY TO ACHIEVE SOME USER CONTROL OVER SIMULATED
C             EXECUTION AND OPERATION FOR TESTING PURPOSES.
C
  130 GOTO(140,10),STAT6
  140 STAT1 = 10 ** RTRN
  150 BRNCH = 1
      RETURN
C
CC          (6) IF LIGHT PEN DETECTED "FORECAST", RESET STAT2
C
  200 IF(PEN(4) - 44)210,201,210
  201 GOTO(202,203),STAT2
  202 STAT2 = 2
      CALL INCL(IGDS1,45)
      STAT7 = STAT5
      BRNCH = 4
      RETURN
  203 STAT2 = 1
      CALL OMIT(IGDS1,45)
      CALL OMIT(IGDS3)
      CALL OMIT(IGDS1,39)
      GOTO 10
C
CC          (7) IF LIGHT PEN DETECTED "SHUTDOWN", BRANCH TO DOWN
C
  210 IF(PEN(4) - 46)220,211,220
  211 BRNCH = 2
      RETURN
C
CC          (8) IF LIGHT PEN DETECTED "APPHA", BRANCH TO PRED1
C
  220 IF(PEN(4) - 40)230,221,230
  221 BRNCH = 5
      RETURN
C
CC          (9) IF LIGHT PEN DETECTED "T-BASE", BRANCH TO PRED2
C
  230 IF(PEN(4) - 42)240,231,240
  231 BRNCH = 6
      RETURN
C
CC          (10) IF LIGHT PEN DETECTED ON ONE OF VALID PATTERNS, BRANCH
CC              TO HIST.
C
  240 IF(PEN(4) - 19)241,241,10
  241 IF(PEN(4) - 1)10,242,242
```

```
  242 ICRVL = 50 + STAT5
      CALL OMIT(IGDS1,39)
      STAT2 = 1
      CALL OMIT(IGDS1,45)
      CALL OMIT(IGDS1,ICRVL)
      CALL OMIT(IGDS3)
      STAT5 = PEN(4)
      ICRVL = STAT5 + 50
      CALL INCL(IGDS1,ICRVL)
      BRNCH = 3
      RETURN
C
CCCC      THIS ROUTINE SHOULD BE PERMANENTLY CORE-RESIDENT FOR
CCC       MAXIMUM STEADY-STATE SPEED.
C
      END
CC


      SUBROUTINE RECVR
C
CCCC      SUBROUTINE RECVR RECEIVES PRE-PROCESSED INFORMATION
CCCCCC    UNITS ACROSS EXCHANGE INTERFACE.  THE "PRE-PROCESSOR" IS
CCCCCC    A PRE-RECORDED FILE FOR TESTING PURPOSES.
CCCCCCCC
CCCCCCCC       THE "INFORMATION UNIT" CONTAINS THE FOLLOWING
CCCCCCCC  INFORMATION IN STANDARD EBCDIC CHARACTERS IN STANDARD
CCCCCCCC  FORMAT:
CCCCCCCC
CCCCCCCC       (A) FUTURES OPTION DESIGNATION AS THREE CHARACTER
CCCCCCCC  MAXIMUM MNEMONIC, AS TRANSMITTED BY CHICAGO MERCANTILE
CCCCCCCC  EXCHANGE.
CCCCCCCC
CCCCCCCC      (B) A QUALIFIER AS FOLLOWS:
CCCCCCCC         A = ASK
CCCCCCCC         B = BID
CCCCCCCC         V = VOLUME REPORT
CCCCCCCC         BLANK = SALE
CCCCCCCC
CCCCCCCC      (C) NUMERICAL VALUE OF TRANSACTION DESIGNATED BY QUALIFIER
CCCCCCCC  IN DECIMAL, NOT INTEGER, REPRESENTATION.
CCCCCCCC
CCCCCCCC      (D) TIME OF TRANSMISSION OF INFORMATION UNIT IN INTEGER
CCCCCCCC  HOURS ON 24-HOUR CLOCK, AND HUNDREDTHS OF A SECOND INTO
CCCCCCCC  THAT HOUR, IN INTEGER REPRESENTATION.
CCCCCCCC
C-----> DEFINE GLOBAL STORAGE
```

```
C
      INTEGER*4 MSTB1,MSTB2,MSTB3,MSTB4
      COMMON /MSTB/ MSTB1(19),MSTB2(19),MSTB3(19),MSTB4(19)
C
CC       MSTB: MASTER TABLE, CONTAINING CURRENT MONITORED COMMODITIES
CC
CC         MSTB1: NUMBER OF SIGNIFICANT CHARACTERS IN MNEMONIC
CC         MSTB2: THIRD CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED, IF ANY
CC         MSTB3: SECOND CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED
CC         MSTB4: FIRST CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED
C
      INTEGER*4 INDEX,HOURS,HSEC
      REAL*4 TRANS,TIME
      COMMON /INPUT/ HOURS,HSEC,INDEX,TRANS,TIME
C
CC       INPUT: INPUT BUFFER FOR EXCHANGE-INTERFACE
CC
CC         HOURS: HOUR OF DAY OF TRANSMISSION, IN 24-HOUR TIME
CC         HSEC: HUNDRETHS SECONDS INTO HOUR
CC         INDEX: HASH-ADDRESS OF MNEMONIC IN MASTER TABLE
CC         TRANS: REAL REPRESENTATION OF "NUMERICAL VALUE" OF INPUT
CC         TIME: TIME OF TRANSMISSION EXPRESSED IS DISPLAY-ORIENTED INCHES
C
      INTEGER*4 STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
      COMMON /SYSSTE/ STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
C
CC       SYSSTE: SYSTEM STATE INDICATORS
CC
CC         STAT1:INDICATES NUMBER OF TIMES LEFT TO RECEIVE DATA (TEST)
CC         STAT2: INDICATES PREDICTIVE STATE ACTIVE IF EQUAL TO 2
CC         STAT3: INDICATES TYPE OF TRANSACTION ON EXCHANGE INTERFACE
CC            =2: ASK;
CC            =3: BID
CC            =4: SALE
CC            =5: VOLUME REPORT
CC         STAT5: INDICATES MASTER TABLE INDEX OF ACTIVE HISTORICAL OVERLAY
CC         STAT6: INDICATES WHETHER EXCHANGE INTERFACE OPEN OR CLOSED
CC         STAT7: MASTER TABLE INDEX OF ACTIVE PREDICTIVE DISPLAY
C
      INTEGER*4 IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
      COMMON /SYSIO/ IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
C
CC       SYSIO: SYSTEM INPUT/OUTPUT UNIT DESIGNATIONS
CC
CC         IO1: SYSTEM PRINTER
CC         IO2: SYSTEM CARD READER
CC         IO3: EXCHANGE INTERFACE INPUT FILE
C          IO4: GRAPHIC DISPLAY OUTPUT UNIT
```

```
C           I05: RAW INFORMATION RECORDING UNIT (SEQUENTIAL)
C           I06: HISTORICAL GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C           I07: PREDICTIVE GRAPHIC OVERLAY FILE (DIRECT ACCESS)
C           I08: SYSTEM "SAVE" FILE WHEN SHUT DOWN (SEQUENTIAL)
C
      INTEGER*4 CHA,CHB,CHC,NOSIG,HSHCD
      COMMCN /BUFF/ CHA,CHB,CHC,NOSIG,HSHCD
C
C          BUFF: TRANSFER BUFFER FOR HASH-ADDRESS GENERATOR
C
C           CHA: FIRST CHARACTER OF MNEMONIC, IF ANY, LEFT-JUSTIFIED
C           CHB: SECOND CHARACTER OF MNEMONIC, LEFT JUSTIFIED
C           CHC: THIRD CHARACTER OF MNEMONIC, LEFT-JUSTIFIED
C           NOSIG: NUMBER OF SIGNIFICANT CHARACTERS IN MNEMONIC
C           HSHCD: HASH ADDRESS GENERATED FROM MNEMONIC
C
      INTEGER*4 BRNCH
      COMMCN /SYSBRN/ BRNCH
C
C         SYSBRN: SYSTEM BRANCH INDICATOR
C
C           THIS INDICATOR IS SET BY SUBROUTINES TO SHIFT SYSTEM CONTROL
C           AFTER EXECUTION.
C
C
C-----> DEFINE LOCAL STORAGE
      INTEGER*4 BLANK,ASK,BID,VOL
C
C-----> DEFINE CONFIGURATION-DEPENDENT EBCDIC INTEGER REPRESENTATIONS
C
      BLANK = 1077952576
      ASK = -1052753856
      BID = -1035976640
      VOL = -448774080
C
      GOTO(200,100),STAT6
C
C-----> IF IN END-OF-FILE STATUS RETURN TO MONITOR
C
  100 BRNCH = 1
      STAT1 = 0
      RETURN
  200 IF(STAT1)210,210,300
C
C-----> IF ALL TRANSACTIONS DESIRED HAVE BEEN READ, RETURN TO MONITOR
C
  210 GOTO 100
C
```

```
C-----> READ A TRANSACTION FROM THE INTERFACE BUFFER
C
  300 READ(IO3,1000,END=900)CHA,CHB,CHC,STAT3,TRANS,HOURS,HSEC
 1000 FORMAT(3A1,1X,A1,1X,F5.2,1X,I2,1X,I6)
      IF(HOURS.LT.9)GOTO 300
C
C-----> TEST COMMODITY MNEMONIC TO SEE IF BEING ACTIVELY MONITORED.
C
      NOSIG = 3
      IF(CHA.EQ.BLANK)NOSIG = 2
      CALL HASH
      IF(MSTB1(HSHCD).NE.NOSIG)GOTO 300
      IF(CHA.NE.MSTB2(HSHCD))GOTO 300
      IF(CHB.NE.MSTB3(HSHCD))GOTO 300
      IF(CHC.NE.MSTB4(HSHCD))GOTO 300
      INDEX = HSHCD
      STAT1 = STAT1 - 1
C
C-----> CONVERT TRANSACTION MNEMONIC TO INTERNAL REPRESENTATION STATE
C
      IF(STAT3.NE.BLANK)GOTO 310
      STAT3 = 4
      BRNCH = 2
      RETURN
C
  310 IF(STAT3.NE.ASK)GOTO 320
      STAT3 = 2
      BRNCH = 2
      RETURN
C
  320 IF(STAT3.NE.BID)GOTO 330
      STAT3 = 3
      BRNCH = 2
      RETURN
C
  330 IF(STAT3.NE.VOL)GOTO 300
      STAT3 = 5
      BRNCH = 2
      RETURN
C
C-----> IF END-OF-FILE OCCURS, RESET SYSTEM STATE # 5
C
  900 STAT6 = 2
      GOTO 100
C
C         THIS ROUTINE SHOULD BE PERMANENTLY CORE-RESIDENT FOR
C         MAXIMUM STEADY-STATE SPEED.
C
```

```
      END
C
C
      SUBROUTINE UPDTE
C
CC
CC        SUBROUTINE UPDTE UPDATES RELEVENT HISTORICAL AND
CC        PREDICTIVE GRAPHIC OVERLAY DIRECT ACCESS FILES EACH TIME
CC        AND INFORMATION UNIT IS RECEIVED ACROSS THE EXCHANGE
CC        INTERFACE.  IF THE INFORMATION UNIT PERTAINS TO A
CC        FUTURES OPTION WHICH IS CURRENTLY 25ING DISPLAYED,
CC        THEN THE DISPLAY IS TRANSFERRED TO THE GRAPHIC DISPLAY
CC        DEVICE AND LEFT THERE.
CC
C-----DEFINE GLOBAL STORAGE
C
      INTEGER*4 IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
      COMMON /SYSIO/ IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8
C
CC        SYSIO: SYSTEM INPUT/OUTPUT UNIT DESIGNATIONS
CC
CC          IO1: SYSTEM PRINTER
CC          IO2: SYSTEM CARD READER
CC          IO3: EXCHANGE INTERFACE INPUT FILE
CC          IO4: GRAPHIC DISPLAY OUTPUT UNIT
CC          IO5: RAW INFORMATION RECORDING UNIT (SEQUENTIAL)
CC          IO6: HISTORICAL GRAPHIC OVERLAY FILE (DIRECT ACCESS)
CC          IO7: PREDICTIVE GRAPHIC OVERLAY FILE (DIRECT ACCESS)
CC          IO8: SYSTEM "SAVE" FILE WHEN SHUT DOWN (SEQUENTIAL)
C
      INTEGER*4 HOURS,HSEC,INDEX
      REAL*4 TRANS,TIME
      COMMON /INPUT/ HOURS,HSEC,INDEX,TRANS,TIME
C
CC        INPUT: INPUT BUFFER FOR EXCHANGE-INTERFACE
CC
CC          HOURS: HOUR OF DAY OF TRANSMISSION, IN 24-HOUR TIME
CC          HSEC: HUNDRETHS SECONDS INTO HOUR
CC          INDEX: HASH-ADDRESS OF MNEMONIC IN MASTER TABLE
CC          TRANS: REAL REPRESENTATION OF "NUMERICAL VALUE" OF INPUT
CC          TIME: TIME OF TRANSMISSION EXPRESSED IS DISPLAY-ORIENTED INCHES
C
      INTEGER*4 MSTB1,MSTB2,MSTB3,MSTB4
      COMMON /MSTB/ MSTB1(19),MSTB2(19),MSTB3(19),MSTB4(19)
C
CC        MSTB: MASTER TABLE, CONTAINING CURRENT MONITORED COMMODITIES
```

```
C
C          MSTB1: NUMBER OF SIGNIFICANT CHARACTERS IN MNEMONIC
C          MSTB2: THIRD CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED, IF ANY
C          MSTB3: SECOND CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED
C          MSTB4: FIRST CHARACTER OF MNEMONIC, RIGHT-JUSTIFIED
C
      INTEGER*4 STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
      COMMON /SYSSTE/ STAT1,STAT2,STAT3,STAT5,STAT6,STAT7
C
C        SYSSTE: SYSTEM STATE INDICATORS
C
C          STAT1:INDICATES NUMBER OF TIMES LEFT TO RECEIVE DATA (TEST)
C          STAT2: INDICATES PREDICTIVE STATE ACTIVE IF EQUAL TO 2
C          STAT3: INDICATES TYPE OF TRANSACTION ON EXCHANGE INTERFACE
C              =2: ASK;
C              =3: BID
C              =4: SALE
C              =5: VOLUME REPORT
C          STAT5: INDICATES MASTER TABLE INDEX OF ACTIVE HISTORICAL OVERLAY
C          STAT6: INDICATES WHETHER EXCHANGE INTERFACE OPEN OR CLOSED
C          STAT7: MASTER TABLE INDEX OF ACTIVE PREDICTIVE DISPLAY
C
      INTEGER*4 VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      INTEGER*4 KEY1,KEY2,KEY3,KEY4,KEY5
      INTEGER*4 ATTN
      REAL*4 TO,DXDT,SLO,VLO
      COMMON /GRAPH/ VIDEO,NULL,I2250,IGDS1,IGDS2,IGDS3
      COMMON /GRAPH/ ATTN,KEY1,KEY2,KEY3,KEY4,KEY5,TO,DXDT,SLO,VLO
C
C        GRAPH: CONTAINS PARAMETERS SPECIFIC TO GRAPHIC DISPLAY SYSTEM
C
C          VIDEO: GRAPHIC SUBROUTINE PACKAGE "GSPNAME"
C          NULL: GRAPHIC SUBROUTINE PACKAGE "NULL" VARIABLE
C          I2250: GRAPHIC SUBROUTINE PACKAGE DEVICE NAME
C          IGDS1: RESIDENT GRAPHIC DATA SET ADDRESS
C          IGDS2: HISTORICAL OVERLAY GRAPHIC DATASET ADDRESS
C          IGDS3: PREDICTIVE OVERLAY GRAPHIC DATA SET ADDRESS
C          ATTN: PRIMARY ATTENTION LEVEL BASE ADDRESS
C          KEY (ALL): RELATIVE ADDRESSES OF SPECIFIEC GRAPHIC ORDERS
C                     IN IGDS2 HISTORICAL FILE
C          TO: SCREEN LOCATION OF 9:00 A.M.
C          DXDT: INCHES PER HOUR ON DISPLAY SCREEN
C          SLO: VERTICAL ORIGIN FOR LOWER LIMIT OF SALES ON SCREEN
C          VLO: VERTICAL ORIGIN FOR "0" FOR VOLUME ON SCREEN
C
      INTEGER*4 BLOCK,SUBBLK
      COMMON/BLOCK/BLOCK(522),SUBBLK(5)
C
```

```
C         BLOCK: PRIMARY SYSTEM INPUT/OUTPUT BUFFERS
CCC
CCC         BLOCK: SHARED HISTORICAL AND PREDICTIVE DIRECT-ACCESS
CCC         SUBBLK: RAW HISTORICAL RECORDER OUTPUT BUFFER
C
      INTEGER*4 INC40,INC50,MXBLK6,MXBLK7
      COMMON /DISKIO/ INC40,INC50,MXBLK6,MXBLK7
C
CCC      DISKIO: SPECIFIES DIRECT-ACCESS I/O POINTERS AND GRAPHIC
CCC              DISPLAY BUFFER SIZES:
CCC
CCC         INC40: POINTS TO NEXT RECORD TO BE READ ON I06
CCC         INC50: POINTS TO NEXT RECORD TO BE READ ON I07
CCC         MXBLK6: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I06 FILE
CCC         MXBLK7: NUMBER OF BYTES ALLOCATED IN 2250 DISPLAY TO I07 FILE
CC
C-----DEFINE LOCAL STORAGE
C
      LOGICAL*1 DISP(5)
C
      INTEGER*4 HOLDS(64)
      INTEGER*4 BLOCKS(74)
      INTEGER*4 HOLD(512)
      INTEGER*4 ITRNS
      INTEGER*4 GDALN,KEYT,MIN,SEC
      INTEGER*4 NOLPS
      INTEGER*4 ICRVL,II
C
      REAL*4 YLC1
      REAL*4 RHOLD
      REAL*4 BASE1,BASE2,SCALE1,SCALE2
      REAL*4 BAS1,SCAL1,BAS2,SCAL2
      REAL*4 BETA,BET1,SMO1,SMO2,SMO3,LSTM
      REAL*4 EXSM1(3),          LSTM1,TMBS,ALPHA,LSTM2,ALPH1
      REAL*4 BASE,SCALE,XLOC,YLOC
      REAL*4 AS0,AS1,AS2,                TAU,SULIM,VULIM
C
C-----DEFINE ACCESSED FILES
C
      DEFINE FILE 40(19,522,U,INC40)
      DEFINE FILE 50(19,74,U,INC50)
C
C-----DEFINE RELEVANT EQUIVALENT STORAGE LOCATIONS
C
      EQUIVALENCE (BLOCK(1),BLOCKS(1))
      EQUIVALENCE (BLOCK(11),HOLD(1),HOLDS(1))
      EQUIVALENCE (BLOCK(4),LSTM1)
      EQUIVALENCE (BLOCK(5),TMBS)
```

```
      EQUIVALENCE (GDALN,BLOCK(6))
      EQUIVALENCE (BLOCK(7),ALPHA)
      EQUIVALENCE (BAS1,BLOCK(7)),(SCAL1,BLOCK(8))
      EQUIVALENCE (BLOCK(8),EXSM1(1))
      EQUIVALENCE (BAS2,BLOCK(9)),(SCAL2,BLOCK(10))
      EQUIVALENCE (SUBBLK(5),RHOLD)
      SULIM = SLO + 4.0
C
CC          (1) UPDATE RAW HISTORICAL FILE
C
C                 (A) PLACE DATA IN SUBBLK
      SUBBLK(1) = INDEX
      SUBBLK(2) = STAT3
      SUBBLK(3) = HOURS
      SUBBLK(4) = HSEC
      RHOLD = TRANS
C
CC                (B) WRITE ON SEQUENTIAL FILE
C
      WRITE(I05)SUBBLK
C
C----------> BEGIN UPDATE OF I06 HISTORICAL FILE <----------
C
CC          (2) LOAD GSP FILE INTO BLOCK FROM I06
C
      READ(I06'INDEX)BLOCK
      SCALE1 = SCAL1
      SCALE2 = SCAL2
      BASE1 = BAS1
      BASE2 = BAS2
C
CC          (3) TRANSFER GRAPHIC ORDERS TO GDOA OF IGDS2 IN CORE FROM HOLD
C
      CALL RESET(IGDS2)
      CALL ORGEN(IGDS2,HOLD,GDALN)
C
CCC         (4) UPDATE GRAPHIC ELEMENTS IN IGDS2
CCC
CCC               (A) CALCULATE SCREEN LOCATION OF NEW POINT, AND BRANCH
C
      IF(STAT3 - 5)245,250,900
C
CC          ASSIGN BASE AND SCALE VALUES FOR SALES AND VOLUME DISPLAYS
C
  245 BASE = BAS1
      SCALE = SCAL1
      GOTO 260
```

```
  250 BASE = BAS2
      SCALE = SCAL2
C
CCC         CALCULATE TIME IN LINEAR UNITS FOR DISPLAY
C
  260 TIME = TO +(FLOAT(HOURS - 9) + FLOAT(HSEC) * 2.78E-06)* DXDT
      CALL IDPOS(IGDS2,0.0,0.0)
C
CCC         SET X AND Y LOCATIONS FOR PLOTTED TRANSACTION POINT
C
      XLOC = TIME
      YLOC = (TRANS - BASE)*SCALE
C
CCC         BRANCH ACCORDING TO QUALIFIER IN INFORMATION UNIT
C
      GOTO(900,300,300,300,340),STAT3
  300 YLOC = YLOC + SLO
C
CCC         IF SALES INFORMATION OVERFLOWS AXES, PLOT "O" AT BOUNDARY
C
      IF(YLOC.LE.SULIM)GOTO 301
      CALL PTEXT(IGDS2,'O',1,NULL,NULL,NULL,XLOC,SULIM)
      GOTO 350
C
CCC         IF SALES INFORMATION UNDERFLOWS AXES, PLOT "U" AT BOUNDARY
C
  301 IF(YLOC.GE.SO)GOTO 305
      CALL PTEXT(IGDS2,'U',1,NULL,NULL,NULL,XLOC,SO)
      GOTO 350
  305 GOTO(900,310,320,330),STAT3
C
CCC         IF SALES INFORMATION WAS AN "ASK" THEN PLOT A "+"
C
  310 CALL PTEXT(IGDS2,'+',1,NULL,NULL,NULL,XLOC,YLOC)
      GOTO 350
C
CCC         IF SALES INFORMATION WAS A "BID", THEN PLOT A "-"
C
  320 CALL PTEXT(IGDS2,'-',1,NULL,NULL,NULL,XLOC,YLOC)
      GOTO 350
C
CCC         IF SALES INFORMATION WAS A SALE, THEN PLOT A POINT
C
  330 CALL PPNT(IGDS2,XLOC,YLOC)
      GOTO 350
C
CCC         IF INFORMATION UNIT WAS A VOLUME REPORT, THEN PLOT A LINE
C           IN VOLUME REGION OF DISPLAY
```

```
C
  340 CALL STPOS(IGDS2,XLOC,1.500)
      YLOC = YLOC + VLO
      CALL PLINE(IGDS2,XLOC,YLOC)
C
CC          TRANSFER THE DISPLAY TO THE 2250 BUFFER TO BE READ
C
  350 CALL EXEC(IGDS2)
C
CC          UPDATE TEXTUAL INFORMATION IN DISPLAY, IGDS2
C
      CALL IDPOS(IGDS2,0.0,0.0)
      GOTO(900,210,220,230,360),STAT3
  210 CALL PTEXT(IGDS2,'ASK ',4,NULL,KEY1,3,9.925,9.800)
      GOTO 240
  220 CALL PTEXT(IGDS2,'BID ',4,NULL,KEY1,3,9.925,9.800)
      GOTO 240
  230 CALL PTEXT(IGDS2,'SALE',4,NULL,KEY1,3,9.925,9.800)
C             (B) UPDATE PRICE IF BID,ASK,SALE
  240 CALL BCNV(TRANS,DISP,102,5,2)
      CALL PTEXT(IGDS2,DISP,5,NULL,KEY2,3,9.835,9.110)
C             (C) UPDATE TIME IF ASK,BID,SALE
      CALL BCNV(HOURS,DISP,103,2)
      CALL PTEXT(IGDS2,DISP,2,NULL,KEY3,3,9.310,8.500)
      SEC = IFIX(FLOAT(HSEC) * 0.01)
      MIN = SEC / 60
      SEC = SEC - MIN * 60
      CALL BCNV(MIN,DISP,103,2)
      CALL PTEXT(IGDS2,DISP,2,NULL,KEY4,3,9.720,8.500)
      CALL BCNV(SEC,DISP,103,2)
      CALL PTEXT(IGDS2,DISP,2,NULL,KEY5,3,10.098,8.500)
C           (7) CHECK TO SEE IF GRAPHIC DISPLAY ACTIVE, THIS PATTERN;IF SO
CC
CC          TRANSFER  IGDS2 TO BUFFER AND INCLUDE IF DISPLAY IS ACTIVE
C
  360 IF(STAT5 - INDEX)370,365,370
  365 CALL INCL(IGDS2)
C
CC          (8) WRITE ALL GRAPHIC DATA BACK ONTO I06 SEGMENT
CC
CCC             (A) MOVE IGDS2 BACK INTO "HOLD"
C
  370 CALL GSPRD(IGDS2,HOLD,MXBLK6,2,GDALN)
      IF(GDALN)38C,380,390
  380 GDALN = MXBLK6
C
CC              (B) WRITE BLOCK BACK ONTO DISK I06
C
```

```
  390 IF(GDALN - MXBLK6)400,392,392
C
C          IF FILE OVERFLOW IS ABOUT TO TAKE PLACE IN IGDS2,
C          WRITE A WARNING ON IO1, SYSTEM PRINTER, AND CONTINUE
C
  392 WRITE(IO1,1000)INDEX,GDALN
 1000 FORMAT(/,' OVERFLOW WARNING ON IO6, SECTOR ',I2,', TOTAL BYTES BEF
     1ORE TRUNCATION= ',Z8,/)
      GDALN = MXBLK6
  400 WRITE(IO6'INDEX)BLOCK
C
C----------> END UPDATE OF IO6 HISTORICAL FILE <----------
C
C
C          UPDATE TIME LINE IN RESIDENT GRAPHIC DISPLAY
C
      ICRVL = 20
      CALL IDPOS(IGDS1,0.0,0.0)
      YLOC = 1.5
      DO 500 II = 1,8,1
      ICRVL = ICRVL + 1
      YLC1 = YLOC + 0.25
      CALL PSGMT(IGDS1,TIME,YLC1,TIME,YLOC,ICRVL,NULL,3)
  500 YLOC = YLOC + 0.50
      YLOC = 6.0
      DO 550 II = 1,8,1
      ICRVL = ICRVL + 1
      YLC1 = YLOC + 0.25
      CALL PSGMT(IGDS1,TIME,YLC1,TIME,YLOC,ICRVL,NULL,3)
  550 YLOC = YLOC + 0.50
      CALL IDPOS(IGDS1,0.0,0.0)
      CALL EXEC(IGDS1)
      CALL INCL(IGDS1)
C
C----------> BEGIN UPDATE OF IO7 PREDICTIVE FILE <----------
C
      IF(STAT3 - 5)605,900,605
C
C          (9) UPDATE PREDICTIVE BLOCK
C
C
C          READ PREDICTIVE FILE INTO CORE STORAGE
C
  605 READ(IO7'INDEX)BLOCKS
      LSTM = LSTM1
      LSTM1 = TIME
C
C          CALCULATE MODIFIED ALPHA
```

```
C
      ALPH1 = ALPHA ** ((TIME-LSTM)*TMBS)
      IF(ALPH1.GT.0.999)ALPH1 = 0.999
      IF(ALPH1.LT.0.001)ALPH1 = 0.001
      BET1 = 1.0 - ALPH1
C
CCC         UPDATE RECURSIVE SMOOTHING STATISTICS
C
      EXSM1(1) = ALPH1 * TRANS + BET1 * EXSM1(1)
      EXSM1(2) = ALPH1 * EXSM1(1) + BET1 * EXSM1(2)
      EXSM1(3) = ALPH1 * EXSM1(2) + BET1 * EXSM1(3)
C
CCCC        CALCULATE CO-EFFICIENTS OF PREDICTIVE EQUATION, WHICH IS
CCCC        PRESUMED TO BE A QUADRATIC MODEL
C
      NOLPS = IFIX((TO + DXDT * 4.0 - TIME) *  5.0)
      AS0 = 3.0 * EXSM1(1) - 3.0 * EXSM1(2) + EXSM1(3)
      AS1 = (ALPH1/(2.0 * BET1)) * ((6.0 - 5.0 * ALPH1) * EXSM1(1) - 2.0
     1* (5.0 - 4.0 * ALPH1) * EXSM1(2) + (4.0 - 3.0 * ALPH1) * EXSM1(3))
      AS2 = (ALPH1 ** 2/BET1 ** 2) * (EXSM1(1) - 2.0 * EXSM1(2) + EXSM1(
     13))
      TAU = 0.0
C
CCCC        RESET IGDS3 AND SET A DUMMY GRAPHIC ORDER IN OVERLAY
CCCC        FILE, IN CASE PREDICTIVE LINE OVERFLOWS OR UNDERFLOWS AXES
C
      CALL RESET(IGDS3)
      CALL STPOS(IGDS3,0.0,0.0)
C
CCCC        GENERATE PREDICTIVE TIME LINE, OMMITTING GENERATION OF
CCCC        A GRAPHIC ORDER IF POINT WILL NOT BE IN AXIS REGION
C
      DO 664 II = 1,NOLPS,1
      TAU = TAU + 0.2
      XLOC = TIME + TAU
      YLOC = SLO + (AS0 + TAU * AS1 + TAU ** 2 * 0.5000 * AS2 - BASE1)
     1 * SCALE1
      IF(SULIM - YLOC)663,662,662
  662 IF(YLOC - SLO)663,664,664
  664 CALL PPNT(IGDS3,XLOC,YLOC,NULL,KEYT)
C
CCCC        TRANSFER IGDS3 TO 225/ BUFFER TO BE READ INTO OVERLAY
CCCC        DIRECT ACCESS UNIT VIA CORE
C
  663 CALL EXEC(IGDS3)
      IF(STAT7 - STAT5)683,681,683
  681 GOTO(683,682),STAT2
  682 CALL INCL(IGDS3)
```

```
  683 CALL GSPRD(IGDS3,HOLDS,MXBLK7,2,GDALN)
      IF(GDALN)684,684,685
  684 GDALN = MXBLK7
  685 IF(GDALN - MXBLK7)686,687,687
C
CC          IF PREDICTIVE FILE IS ABOUT TO OVERFLOW, WRITE A MESSAGE TO
CC          IO1, THE SYSTEM PRINTER, AND CONTINUE
C
  687 WRITE(IO1,1010)INDEX,GDALN
 1010 FORMAT(/,' OVERFLOW WARNING ON IO7, SECTOR ',I2,', TOTAL BYTES BEF
     1ORE TRUNCATION= ',Z8,/)
      GDALN = MXBLK7
C
CC          TRANSFER PREDICTIVE OVERLAY FILE BACK TO DIRECT ACCESS
C
  686 WRITE(IO7'INDEX)BLOCKS
C
C----------> END UPDATE OF IO7 PREDICTIVE FILE <----------
CC
CC
CC         (10) RETURN TO MAINLINE
C
  900 RETURN
C
CC          THIS ROUTINE SHOULD BE PERMANENTLY CORE-RESIDENT FOR
CC          MAXIMUM STEADY-STATE SPEED.
C
      END
//*
//*
//*       THE FOLLOWING INPUT TO THE LINKAGE EDITOR DEFINES:
//*
//*            A. THE GRAPHIC SUBROUTINE LIBRARY ON SYSLIB;
//*
//*            B. THE DYNAMIC OVERLAY STRUCTURE TO CONSERVE CORE.
//*
//*
//LINK.SYSIN DD *
   INCLUDE SYSLIB(IHCGSP03)
  ENTRY COMMON
   INSERT COMMON,HASH
  OVERLAY ONE
  INSERT BIRTH
  OVERLAY TWO
  INSERT INIT1
  OVERLAY TWO
  INSERT INIT2
  OVERLAY TWO
```

```
  INSERT INIT3
  OVERLAY TWO
  INSERT INIT4
  OVERLAY TWO
  INSERT INIT5
  OVERLAY TWO
  INSERT INIT6
  OVERLAY ONE
  INSERT TBASE
  OVERLAY ONE
  INSERT PRED
  OVERLAY ONE
  INSERT HIST
  OVERLAY ONE
  INSERT ALPHA
  OVERLAY ONE
  INSERT MNITR,RECVR,UPDTE
//*
//*       THE FOLLOWING IS INPUT TO THE COMMODITY MONITOR
//*
//GO.SYSIN DD *
06,05,10,20,30,40,50,60,1,05/25/70       I/O DEFINITION CARD
 EG,2,FEBRUARY, 1970 EGGS              ,50.00,53.00.1300.
+0.22E+01 +0.22E+01 +0.22E+01 0.800 005
 PK,2,MAY, 1970 IDAHO POTATOES     , 5.60, 6.00, 900.
+0.22E+01 +0.22E+01 +0.22E+01 0.800 005
PBQ,3,AUGUST, 1970 PORK BELLIES    ,40.00,43.00, 200.
+0.22E+01 +0.22E+01 +0.22E+01 0.800 005
LHM,3,JUNE, 1970 LIVE HOGS          ,27.00,28.00,  70.
+0.22E+01 +0.22E+01 +0.22E+01 0.800 005
LCQ,3,AUGUST, 1970 LIVE CATTLE     ,30.50,31.50, 200.
+0.22E+01 +0.22E+01 +0.22E+01 0.800 005
//*
//*       DEFINE THE SIMULATED EXCHANGE-INTERFACE FILE
//*
//GO.FT10F001 DD DSN=S0445.INTFCE,DISP=(OLD,PASS),UNIT=2314,
//   VOL=SER=LINDA,DCB=(LRECL=21,BLKSIZE=2100,RECFM=FB)
//*
//*       DEFINE THE GRAPHIC DISPLAY DEVICE
//*
//GO.FT20F001 DD UNIT=(2250-1)
//*
//*       DEFINE THE SEQUENTIAL RAW INFORMATION UNIT RECORDER FILE
//*
//GO.FT30F001 DD DSN=&RAWHST,SPACE=(CYL,(1,1)),
//    DISP=(NEW,DELETE),UNIT=SYSDA
//*
//*       DEFINE DIRECT-ACCESS HISTORICAL GRAPHIC OVERLAY FILE
```

```
//*
//GO.FT40F001 DD DSN=&GSPHST,SPACE=(2088,19),
//    DISP=(NEW,DELETE),UNIT=SYSDA
//*
//*       DEFINE DIRECT-ACCESS PREDICTIVE GRAPHIC OVERLAY FILE
//*
//GO.FT50F001 DD DSN=&GSPPRD,SPACE=(296,19),
//    DISP=(NEW,DELETE),UNIT=SYSDA
//*
//*       DEFINE SEQUENTIAL FILE TO SAVE MASTER TABLE "MSTB"
//*
//GO.FT60F001 DD DSN=&SAVE2,SPACE=(CYL,(1,1)),
//    DISP=(NEW,DELETE),UNIT=SYSDA
//*
//*       DEFINE SEQUENTIAL FILE TO SAVE HISTORICAL GRAPHIC OVERLAY FILE
//*
//GO.FT60F002 DD DSN=&SAVE4,SPACE=(2088,19),
//    DISP=(NEW,DELETE),UNIT=SYSDA
//*
//*       DEFINE SEQUENTIAL FILE TO SAVE PREDICTIVE GRAPHIC OVERLAY FILE
//*
//GO.FT60F003 DD DSN=&SAVE5,SPACE=(296,19),
//    DISP=(NEW,DELETE),UNIT=SYSDA
```

BIBLIOGRAPHY

Western Electric Corporation, Description and Adjustments of the Type-wheel Tape Printer, Teletype Bulletin No. 134, January, 1934.

Brown, R. G., Smoothing, Forecasting and Prediction of Discrete Time Series, Prentice-Hall, 1963.

Belveal, L., Charting Commodity Market Price Behavior, Commodity Press, 1969.

IBM Corporation, System/360 Operating System, Graphic Subroutine Package for FORTRAN IV, COBAL, and PL/1, Form C27-6932-3, November, 1968.

IBM Corporation, System/360 Operating System, Linkage Editor and Loader, Form C28-6538-8, November, 1969.

IBM Corporation, System/360 FORTRAN IV Language, Form C28-6515-7, October, 1968.

IBM Corporation, System/360 Operating System, FORTRAN IV (G and H) Programmer's Guide, Form C28-6817-1, July, 1969.

Feldman, J. and Gries, D., "Translator Writing Systems", Communications of the ACM, Vol. 11, p. 77-113, February, 1968.

Chicago Mercantile Exchange, Public information pamphlets:
"How Commodities are Bought and Sold"
"Trading in Tomorrows"
"Before You Speculate..."
"Trading Techniques for the Commodity Speculator"

Ultronics Systems Corporation, "The Information Explosion" (Technical sales brochure), January, 1969

Morris, R., "Scatter Storage Techniques", Communications of the ACM, Vol. 11, p. 38-44, January, 1968.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Asst. Prof. G. L. Barksdale, Code 53Bv Department of Mathematics Naval Postgraduate School Monterey, California 93940	1
4.	LTJG Brian Smith Bentley, USNR 30 North Main St., Ipswich, Massachusetts 01938	1

Blank

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1 ORIGINATING ACTIVITY *(Corporate author)*
Naval Postgraduate School
Monterey, California 93940

2a. REPORT SECURITY CLASSIFICATION
Unclassified

2b. GROUP

3 REPORT TITLE
A Proposed Real-time Monitor and Data Analyzer for an American Commodity Exchange

4 DESCRIPTIVE NOTES *(Type of report and inclusive dates)*
Master's Thesis; June 1970

5 AUTHOR(S) *(First name, middle initial, last name)*
Brian Smith Bentley

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
June 1970	138	11

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*
d.	

10 DISTRIBUTION STATEMENT
This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

This Thesis considers the application of real-time digital information systems to the field of professional trading of commodities. Pertinent factors involved with the interfacing of a monitor system with a commodity exchange, and with professional trading interests, are considered. Skeletal functional outlines of factual and predictive displays are proposed from the monitoring of a general market environment to the monitoring of specific user positions in that environment. Elements of a monitoring system are implemented on an IBM 360/67 configured with an IBM 2250 Graphic Display Unit operating with OS/MVT. Due to the highly subjective nature of commodity decision-making, a scarcity of relevant published information in this area, and the author's lack of extensive trading experience, this thesis addresses itself only to the acquisition, summarization and presentation of objective information. Any presumption of data interpretation is left to the user.

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Commodity trading						
Real-time monitors						
Commodity exchanges						
Graphic-display information systems						
Commodity prediction, real-time						
Securities						
Commodities, real-time monitors						
Commodity exchange information systems						
Management information systems, real-time						

Thesis
B392
c.1
120639
Bentley
A proposed real-time monitor and data analyzer for an American commodity exchange.